VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2019 Zuzana Miklánková

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

IDENTIFIKAČNÍ A AUTENTIZAČNÍ TECHNOLOGIE
IDENTIFICATION AND AUTHENTICATION TECHNOLOGIES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Zuzana Miklánková**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Petr Dzurenda**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Informační bezpečnost**
Ústav telekomunikací

Studentka: Zuzana Miklánková ***ID:*** 194381
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Identifikační a autentizační technologie

POKYNY PRO VYPRACOVÁNÍ:

Student analyzuje současné RFID technologie a možnosti jejich nasazení v identifikačních a autentizačních systémech. Výstupem bakalářské práce bude funkční implementace atributového přístupového systému složeného z RFID štítků a výkonově omezených zařízení typu čipová karta, Raspberry Pi, Arduino apod. Přístupový systém bude zajišťovat ochranu soukromí uživatelů a bude vyžadovat zapojení více autentizačních prvků do autentizačního procesu.

DOPORUČENÁ LITERATURA:

[1] FINKENZELLER, Klaus. RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication. John Wiley & Sons, 2010.

[2] JUELS, Ari. RFID security and privacy: A research survey. IEEE journal on selected areas in communications, 2006, 24.2: 381-394.

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 27.5.2019

Vedoucí práce: Ing. Petr Dzurenda
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Bakalárska práca sa zaoberá analýzou momentálne dostupných RFID technológií a čipových kariet. Zhrhnuté sú druhy identifikačných a autentizačných rádiových, či kontaktných systémov. Praktická časť porovnáva namerané výkonové výstupy technológií RFID a popisuje implementáciu vlastného viac atribútového autentizačného systému využívajúceho RFID technológie a čipové karty. Autentizačný systém môže zaisťovať napríklad vstup zamestnancov do budovy, pričom budú nutne preukázaní ako aj čipovou kartou, tak i RFID tagmi. Systém tiež zahŕňa prvky pre ochranu súkromia užívateľov, kedy na základe odchytenej komunikácie nie je možné odhaliť identitu užívateľa.

KĽÚČOVÉ SLOVÁ

Autentizácia, autentizačný protokol, RFID, UHF, HF, LF, JavaCard, MULTOS, prístupový systém, ochrana súkromia.

ABSTRACT

The bachelor thesis deals with the analysis of currently available RFID technologies and smart cards. Summarized are types of identification and authentication radio or contact systems. The practical part compares measured RFID functional outputs and describes the implementation of multi-attribute authentication system using RFID technology and smart cards. For example, the authentication system can provide employees with access to the building, necessarily demonstrated by both the smart card and RFID tags.The system also includes user privacy elements, where identity of the user cannot be detected by captured communication.

KEYWORDS

Authentication, authentication protocol, RFID, UHF, HF, LF, JavaCard, MULTOS, access system, privacy protection.

MIKLÁNKOVÁ, Zuzana. *Identifikační a autentizační technologie*. Brno, Rok, 65 s. Bakalárska práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedúci práce: Ing. Petr Dzurenda

VYHLÁSENIE

Vyhlasujem, že som svoju bakalársku prácu na tému „Identifikační a autentizační technologie“ vypracovala samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autorka uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušila autorské práva tretích osôb, najmä som nezasiahla nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomá následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autorky

POĎAKOVANIE

Moje poďakovanie patrí vedúcemu bakalárskej práce, pánovi Ing. Petrovi Dzurendovi za inšpiratívne idey a neustálu podporu.

Brno ....

podpis autorky

Tato práce vznikla jako součást klíčové aktivity KA6 - Individuální výuka a zapojení studentů bakalářských a magisterských studijních programů do výzkumu v rámci projektu OP VVV Vytvoření double-degree doktorského studijního programu Elektronika a informační technologie a vytvoření doktorského studijního programu Informační bezpečnost, reg. č. CZ.02.2.69/0.0/0.0/16_018/0002575.

EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Projekt je spolufinancován Evropskou unií.

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic
http://www.six.feec.vutbr.cz

POĎAKOVANIE

Výskum popísaný v tejto bakalárskej práci bol realizovaný v laboratóriách podporených projektom SIX; registračné číslo CZ.1.05/2.1.00/03.0072, operačný program Výzkum a vývoj pro inovace.

Brno ....

podpis autorky

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

Obsah

Úvod **12**

1 Teoretická časť študentskej práce **13**
1.1 RFID technológie 13
1.2 RFID tag 14
1.2.1 Štandardizácia 14
1.2.2 Rozdelenie podľa operačnej frekvencie 15
1.2.3 Rozdelenie podľa hardvérového zloženia 16
1.2.4 Rozdelenie podľa spôsobu napájania 21
1.3 Kryptografia na RFID 22
1.3.1 Symetrická kryptografia 22
1.3.2 Asymetrická kryptografia 23
1.4 Súkromie a bezpečnosť 25

2 Praktická časť študentskej práce **27**
2.1 Pridelený hardvér a merania 27
2.1.1 LF systém 27
2.1.2 HF systém 27
2.1.3 UHF systém 30
2.2 Meranie kryptografických primitív 36
2.3 Autentizačný protokol 37
2.3.1 Architektúra autentizačného systému 38
2.3.2 Obecná schéma 39
2.3.3 Výber hardvéru a obmedzenia s ním spojené 42
2.3.4 Implementácia autentizačného protokolu 45

Záver **57**

Literatúra **59**

Zoznam skratiek **62**

Zoznam príloh **64**

A Obsah priloženého CD **65**

Zoznam obrázkov

1.1 Čiarový kód UPC-A 13
1.2 Vzťah tag - čítačka - aplikácia 14
1.3 Architektúra čipovej karty 17
1.4 Mikroprocesorový tag 20
1.5 Kryptografia za použitia symetrického kľúča 22
1.6 Kryptografia za použitia asymetrického kľúča 24
2.1 Maximálny dosah každého tagu pri sile čítačky 10 dBm a 30 dBm . . 33
2.2 Úspešnosť identifikácie UHF systému č. 1 34
2.3 Úspešnosť identifikácie UHF systému č. 2 35
2.4 Rýchlosť načítania viacerých UHF tagov zároveň 35
2.5 Rýchlosť AES na čipových kartách 36
2.6 Rýchlosť násobenia bodov na eliptickej krivke na čipovej karte Java Card 37
2.7 Navrhovaný scénar autentizačného systému 38
2.8 Registračná časť autentizačného protokolu 39
2.9 Autentizačná časť 41
2.10 Blokové schéma 3DES 43
2.11 Proces komunikácie medzi JCRE a appletom, JavaCard 47

Zoznam tabuliek

1.1 Triedy tagov podľa EPC 15
1.2 Triedy tagov podľa ISO18000 15
1.3 Štruktúra APDU príkazu 18
1.4 Štruktúra APDU odpovede 18
2.1 Tagy použité na meranie HF RFID systému 28
2.2 Organizovaný sled bajtov posielaný z počítača čítačke na jej ovládanie 31
2.3 Tagy použité na meranie UHF RFID systému 32
2.4 Prehľad použitých programovacích jazykov a vývojových prostredí . . 45
2.5 Priemerné trvanie hlavných častí autentizácie 56

Zoznam výpisov

2.1 Postup prípravy prostredia pre komunikáciu s HF čítačkou 28
2.2 Výpis z autentizácie čipovej karty MIFARE DESFire 30
2.3 Funkcia tagQuery(), dodatočne pridaná do modulu UHFReader18 . . 31
2.4 Výpis skriptu na testovanie UHF systému 34
2.5 Časť výpisu z JCAlgTest 44
2.6 Metóda na hešovanie pomocou SHA1, JavaCard. 46
2.7 Operácia SHA-1 na čipovej karte MULTOS 48
2.8 Nastavenie aid a dir 48
2.9 Násobenie bodu na eliptickej krivke, MULTOS 50
2.10 Nastavenie UHF čítačky 50
2.11 Výpis z interaktívneho skriptu registrácie - generovanie dát SAMu . . 51
2.12 Verejný ECDH kľúč v ASN.1 formáte 52
2.13 Výpis z interaktívneho skriptu registrácie - zadanie ID SAMu 52
2.14 Výpis z interaktívneho skriptu registrácie - UHF časť 53
2.15 Servisný súbor servisy authentication.service 54
2.16 Čiastkový výpis výstupu servisy authentication.service 54

Úvod

V posledných rokoch sa vo svete rozširuje RFID (Radio Frequency Identification). Ide o technológiu využívajúcu rádiové vlny k bezkontaktnej automatickej identifikácií tovaru, ľudí či zvierat na rôzne vzdialenosti. Za pomerne malú finančnú investíciu je možné vybudovať robustný a spoľahlivý systém na kontrolu tovaru, či na spracovanie dochádzky zamestnancov. Neustále pribúdajú výrobcovia takýchto riešení a povedomie o RFID sa rozširuje. Prístupové systémy nachádzajú využitie všade, kde je nutné zabezpečiť individuálny vstup do miestnosti alebo budovy.

V súčastnej dobe existuje mnoho aplikačných scenárov, v ktorých je RFID technológia nasadená. RFID tagy sú jednoducho implantovateľné do zvierat a umožňujú ich jednoduchšiu identifikáciu v prípade ich straty. Je možné nimi sledovať batožinu na letiskách, či poštové zásielky a monitorovať liečebnú históriu pacientov v nemocniciach. Pre využitie v prírode je možné nasadiť odolné systémy, ktoré sú schopné vzdorovať vplyvom počasia a ich maximálna komunikačná vzdialenosť sa môže vyšplhať k desiatkam či stovkám metrov.

Stále však existujú pomerne veľké medzery v bezpečnosti RFID systémov a v ich využití v reálnych autentizačných schémach, ktoré ochraňujú súkromie užívateľa a bezpodmienečne zaručujú jeho identitu.

Bakalárska práca Identifikačné a Autentizačné technológie sa zaoberá základmi RFID systémov, hovorí o ich druhoch a odlišnostiach. Praktická časť experimentálne porovnáva odlišnosti RFID systémov a prehľadne popisuje ich reálne schopnosti. Hlavným výstupom tejto práce je návrh a implementácia viacatribútového autentizačného protokolu zaisťujúceho súkromie užívateľov.

1 Teoretická časť študentskej práce

V teoretickej časti je popísaný úvod do RFID technológií, ich štandardizácia, rozdelenie a využitie. V stručnosti sú popísané čipové karty spoločnosti MIFARE, JavaCard a MULTOS. Na záver sú spomenuté druhy kryptografie, ktoré môžu byť využiteľné v oblasti RFID.

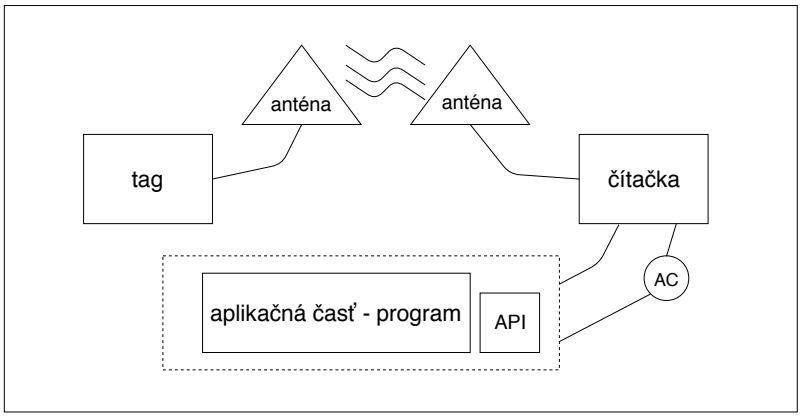
1.1 RFID technológie

Technológia RFID sa začala rozvíjať v 80-tych rokoch dvadsiateho storočia, pre nedostačujúce funkcionality dovtedajších čiarových kódov. Tie boli vytvorené na kódovanie informácií ako napríklad produktové alebo sériové číslo výrobku. Spôsob označovania čiarovými kódmi je vizuálny a z toho dôvodu sú čítateľné ako aj človekom, tak aj strojovo-čítateľné, avšak len za predpokladu priamej viditeľnosti na daný kód. Najznámejší typ čiarového kódu, takzvaný UPC-A (Universal Product Code type A) symbol, viď 1.1, reprezentuje dáta lineárnymi čiarami, kde nositeľmi danej informácie boli hrúbka a vzialenosť od seba.

Obr. 1.1: Čiarový kód UPC-A

Keďže RFID namiesto vizuálneho prenesenia informácie prenáša informáciu na základe rádiových vĺn, informácia nie je priamo čitateľná človekom, avšak strojovo-čitateľná je aj bez priameho zorného dosahu. Ďalším dôležitým bodom je rýchlosť čítania, keď pri využití RFID technológie, môže byť v jednu dobu súčasne načítaných viacero tagov (množsvá sa môžu líšiť v závislosti na type RFID systému).

Táto technológia obsahuje tri oddelené funkčné jednotky. Prvou je tag, nesúci dáta, pomocou ktorých sa identifikuje u čítačky, ktorá je druhou funkčnou jednotkou RFID sústavy. Treťou a poslednou časťou je program - databáza - ktorý spracováva dáta získané z tagu pre ďalšie použitie, prípadne ich ďalej využíva sám. Schéma je vynesená do obrázku 1.2, kde AC (Alternating Current) je striedavý prúd a API (Application Programming Interface) je programová podpora.[1]



Obr. 1.2: Vzťah tag - čítačka - aplikácia

1.2 RFID tag

Tag, ako nositeľ informácie môže nadobúdať rôznych foriem spracovania. Do úvahy treba brať operačnú frekvenciu tagu, formu jeho napájania, či rozmer.[14]

1.2.1 Štandardizácia

Pre RFID technológiu existuje mnoho štandardov, ktoré by mali pomáhať v jednoduchšej implemetácii a rýchlejšiemu vývoju danej technológie. Dvoma najdôležitejšími štandardizačnými organizáciami v tomto sektore sú ISO (International Organization of Standardization)[2] a EPCglobal (Electronics Product Code Global Incorporated)[3]. Každá nahliada na RFID z mierne iného uhlu a určuje podobné, ale miestami odlišné definície. Industriálne štandardy sa často podriaďujú štandardom medzinárodným, pre zvýšené možnosti využitia štandardizovaného výrobku.

EPCglobal

Spoločnosť EPCglobal sa vo svojich štandardoch zaoberá ako aj rádiovým rozhraním, tak aj dátovou štruktúrou tagu. Zavádza pojem EPC (Electronic Product Code), čo je vlastne univerzálny a celosvetovo jedinečný identifikačný kód tagu. RFID technológiu predurčuje na správu zásobovacích reťazcov, na značenie tovarov, teda na systémovú náhradu barkódov. EPC delí tagy podľa možnosti čítania a zápisu, či zdroja energie, viď tab. 1.1. Tag podľa EPC môže byť aj multištandardný.[4]

ISO

Štandardy spoločnosti ISO sa v mnohých štátoch berú ako normatívne, teda ich váha býva najvyššia. Spolu s IEC (International Electrotechnical Commission) navrhujú, udržujú a zjednocujú technické štandardy. Všeobecne RFID spadá pod štan-

Tab. 1.1:

EPC trieda	kategória tagu	popis
trieda 0	read-only	najjednoduchšie tagy
trieda 1	write once, read many	jeden krát zapisovateľné tagy
trieda 2	zapisovateľné	tagy s možnosťou opakovaného zápisu
trieda 3	semi-pasívne	môžu obsahovať rôzne senzory
trieda 4	aktívne tagy	vedia vysielať aj mimo dosahu čítačky

dard ISO/IEC 18000, ktorý sa skladá z viacerých kategórií podľa vlnovej komunikačnej frekvencie tagu, viď. tab 1.2. Okrem frekvencie sa zaoberá moduláciou, kódovaním dát, bitovou rýchlosťou prenosu či frekvenciou prípadne obsiahnutého mikroprocesora.[4]

Tab. 1.2: Triedy tagov podľa ISO18000

ISO 18000 štandard	frekvencia	spektrum
ISO/IEC 18000-2	pod 135 kHz	nízka frekvenzia
ISO/IEC 18000-3	13.56 MHz	vysoká frekvencia
ISO/IEC 18000-4	2.45 GHz	mikrovlny
ISO/IEC 18000-6	od 860 MHz do 960 MHz	ultra vysoká frekvencia
ISO/IEC 18000-7	433 MHz	ultra vysoká frekvencia

ISO vyčleňuje množstvo iných štandardov pre bezkontaktné identifikačné karty, zaoberajúcich sa rôznymi časťami a triedami systému, napríklad štandardy ISO/IEC 1443 pre karty s väzbou na blízko a ISO/IEC 15693 pre karty s väzbou na diaľku, oba typy operujúce iba na frekvencii 13.56 MHz.[4]

1.2.2 Rozdelenie podľa operačnej frekvencie

RFID tagy sú schopné pracovať na rôznych frekvenciách, avšak vždy totožných s frekvenciou čítačky patriacej k danému tagu. Frekvencia odkazuje na kmitočet rádiových vĺn, ktoré sú využívané na komunikáciu medzi časťami systému. Existujú tri druhy frekvencií využívané v RFID technológii, a to LF (Low Frequency), HF (High Frequency) a UHF (Ultra-High Frequency).[5]

LF tagy

LF tagy komunikujú na nízkych frekvenciách, konkrétne 30 KHz - 300 KHz. Najčastejšie je využívaná frekvencia 125 KHz. Majú veľkú vlnovú dĺžku, konkrétne od 1 do 10 km. Dobre prekonávajú aj tenke kovové povrchy. Sú schopné spomedzi všetkých niesť najmenej dát, čo ich zaraďuje medzi pamäťové tagy. Využívajú sa na základnú identifikáciu produktov, zvierat a ľudí. Rozmery tagu môžu byť od najmensích nálepiek o veľkosti 2 cm po formát platobných kariet. Menej časté sú tagy v tvare kľučeniek a príveskov.[6]

HF tagy

HF tagy operujú na vysokých frekvenciách od 3 do 30 MHz, najčastejšie 13,56 MHz. Vlnová dĺžka na týchto frekvenciách sa pohybuje od 10 do 100 metrov. RFID HF systémy majú operačnú vzdialenosť v jednotkách centimetrov. Pracujú pomerne dobre ak sú umiestnené na kovových objektoch a objektoch s vysokým obsahom vody. Na tejto frekvencii pracujú všetky bezkontaktné čipové karty.[7]

UHF tagy

UHF sú ultra vysoké frekvencie, začínajúce na 300 MHz a siahajuce až po 3 GHz. Rozsah najčastejšie používaných frekvencií na ktorých pracujú UHF tagy zavisí na reguláciách každého štátu. Pre Európu je to 865 - 868 MHz, pre oblasť Severná Amerika 902 - 928 MHz. Krátka vlnová dĺžka UHF technológiu nepredurčuje k efektívnemu prekračovaniu prekážok ako hôr, či väčších objektov. Do istej miery je možný prechod vńn stenami budovi. UHF systémy majú najväčšiu operačnú vzdialenosť, v závislosti od sily čítačky a veľkosti tagu a jeho antény to môžu byť metre, či desiatky metrov.

UHF RFID tagy sú vysoko rozšírené hlavne v logistike na značenie tovaru, a teda na vývoj a štandardizáciu tejto technológie tlačia najmä obchodné spoločnosti. Hlavná štandardizujúca spoločnosť v tejto sekcii je EPCglobal, kvôli jej cieleniu na priemyselné využitie technológie.[7]

1.2.3 Rozdelenie podľa hardvérového zloženia

Aj keď základná štruktúra RFID tagov je vždy príbuzná, existujú v nej isté odlišnosti. Každý tag má iné využitie a tomu je prispôsobený aj výber prvkov jeho štruktúry. Rozhodujúce faktory sú čo najnižšia výrobná cena, najmenšia energetická spotreba a najväčšia modularita systému.

EAS tagy

Najjednoduchšie tagy nesú ako svoje dáta iba jeden bit. Ide o takzvané EAS (Electronic Article Surveillance) tagy - tagy na elektronické sledovanie artiklov. Nesú len informáciu, či sa v danom priestore nejaký tag nachádza alebo nenachádza. Neslúžia teda k indentifikácii, ale len k detekcii výskytu. Zariadenia sú rozšírené najmä v obchodoch, kde slúžia ako obrana proti zlodejom. Skladajú sa z cievky ohraničenej dvoma fóliami, ktoré vytvárajú kondenzátor. Cievka indukuje elektrické napätie z ktorého napája činnosť RFID tagu. Ďalej je tu obsiahnutá malá nevolatilná (napäťovo nezávislá) pamäť, ktorá samotná nesie potrebnú informáciu.[6]

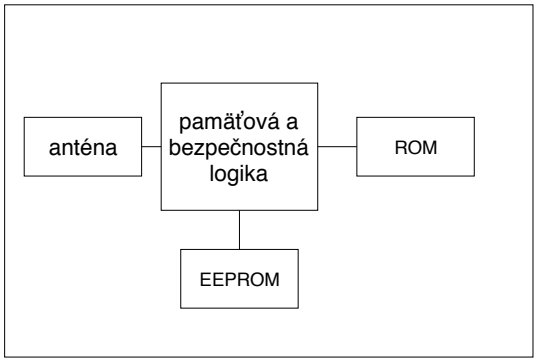
RFID pamäťové tagy

Majú pamäť určenú iba na čítanie, prípadne pamäť, ktorá sa dá zapísať len raz. Nemajú žiadnu výpočetnú jednotku a tak nedokážu vykonavať autentizáciu. Jedinou ich funkciou je pri dostatočnom zdroji elektrickej energie vysielať svoje ID uložené v pamäti.

RFID čipové karty

Pre vykonávanie funkcionálnych inštrukcií na tagu je nutné mať obsiahnutú takisto zapisovateľnú pamäť. Pridaná pamäť je najčastejsie typu EEPROM (Electrically Erasable Programmable Read-Only Memory), ktorá slúži pre uchovanie dát, RAM (Random Access Memory, ktorá slúži na ukladanie dočasných hodnôt vrámci jednej relácie, alebo ich kombináciou. Takisto býva obsiahnutá aj pamäť typu ROM, do ktorej je priamo pri výrobe tagu implementovaná aplikácia na základnú funkčnosť tagu, prípadne samotný operačný systém karty.

Čipové karty môžu obsahovať mikroprocesor, na ktorom sa nachádzajú výrobcom implementované kryptografické a autentizačné funkcie. Schéma čipovej karty je vykreslená na obrázku 1.3.



Obr. 1.3: Architektúra čipovej karty

Čipové karty štandardne komunikujú s terminálom pomocou tzv. APDU (Application Protocol Data Unit) správ. APDU správy zaisťujú komunikáciu medzi čítačkou a kartou a sú štandardizované podľa ISO/IEC 7816. Existujú dve varianty správ APDU, a to príkaz a odpoveď. Príkaz je vyslaný od čítačky smerom ku tagu, hovorí mu, čo od tagu čítačka vyžaduje. Odpoveď smeruje od tagu ku čítačke a obsahuje výsledok počiatočného príkazu. [8]

Štruktúra príkazu sa skladá z povinnej hlavičky a nepovinného tela, viď. tab. 1.3. Štruktúra odpovede je jednoduchšia, viď. tab. 1.4, skladá sa z očakávaných dát a z výstupu vykonaného príkazu. Môže oznamovať napríklad typ chyby, ak sa inštrukcia nepodarí vykonať, alebo správne spracovanie, ak inštrukcia prebehla korektne. [9]

Tab. 1.3: Štruktúra APDU príkazu

pole	**popis**	**počet bajtov**
CLA	identifikácia inštrukčnej triedy	1
INS	inštrukcia z inštrukčnej triedy CLA	1
P1 - P2	paramater č. 1 a č. 2	2
Lc	veľkosť datového pola	0, 1 (pri T=0) alebo 3 (pri T=1)
Dáta	prenášané dáta	len(Lc)
Le	veľkosť očakávaných dát v odpovedi	0, 1 (pri T=0) alebo 3 (pri T=1)

Tab. 1.4: Štruktúra APDU odpovede

pole	**popis**	**počet bajtov**
Dáta	dáta vrátené v odpovedi	$\leq$ len(Le)
SW1 - SW2	stavové slová	2

Najviac rozšíreným výrobcom čipových kariet je NXP Semiconductors so svojou radou MIFARE. Tieto čipové karty nie sú dodatočne doprogramovateľné, jadrové aplikácie do nich vkladá už výrobca. MIFARE zabezpečuje rôzne druhy zabezpečenia a funkcionality, začínajúc od kariet Ultralight.

Tagy **MIFARE Ultralight** sú navrhnuté na čo najrýchlejšiu komunikáciu, pri čo najnižších nákladoch. Základná rada MIFARE Ultralight preto neponúka žiadnu autentizáciu, iba identifikáciu na základe 7 bajtového UID (Unique IDentity). Výrobcom stanovená rýchlosť prenosu je 106 Kbit/s. Tak ako všetky tagy rady MIFARE je z jej pamäte možné čítať, zapisovať na ňu, inkrementovať a dekrementovať daný

blok pamäte. Toto sa netýka UID, ktoré je vypálené výrobcom priamo na kartu a je neprepisovateľné. [8]

O niečo rozvinutejší je tag **MIFARE Classic**, ktorý už zahŕňa nástroje na autentizáciu. Obsahuje výrobcom stanovené 4 bajtové alebo 7 bajtové UID. Pamäť má rozdelenú na sektory, pričom na každý sektor pripadajú dva kľúče, pre trojnásobnú vzájomnú autentizáciu čítačky s kartou, k obsahu pamäte je teda možné pristúpiť až po úspešnej autentizácií. Autentizácia je tu zabezpečená prúdovou šifrou CRYPTO1, ktorá bola prelomená.[10]

Tretí typ tagu z rodiny MIFARE je **DESfire**. Šifrovanie prenesených dát je založené na blokových šifrách 3DES (Triple Data Encryption Standard), alebo AES (Advanced Encryption Standard). Takisto ako Classic podporuje trojnásobnú vzájomnú autentizáciu čítačky s kartou a nutnosť autentizácie pred čítaním jeho pamäte. DESfire je momenálnym štandardom bezpečnosti vrámci rodiny MIFARE, pretože jeho autentizácia ako jediná nebola prelomená (týka sa prevedenia EV2, na prevedenie EV1 bol úspešne prevedný útok postrannými kanálmi - meraním prúdovej spotreby čipu). Jeho UID je dlhé 7 bajtov, celá pamäť je o veľkosti 2 kB alebo 4 kB alebo 8 kB. Celý tag je chránený jedným hlavným kľúčom.[11]

RFID programovateľné čipové karty

Vyššie spomenutá skupina čipových kariet má od výrobcu implementované funkcie a aplikácie, ktoré už ďalej nejde rozširovať, iba využívať. Druhá skupina je určená na doprogramovanie a prispôsobenie si čipovej karty podľa seba, ide o programovateľné čipové tagy. K týmto kartám tradične výrobcobia ponúkajú SDK (Software Development Kit), ktorý obsahuje podporu pre implementovanie vlastných aplikácií.

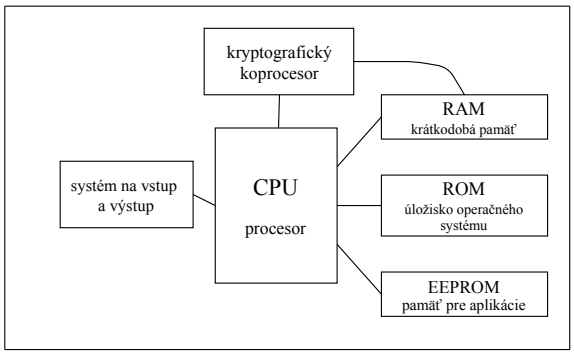
Ich mikroprocesor je priamo programovateľný. Môžu obsahovať aj pomocný koprocesor, na ktorom pre väčšiu rýchlosť prebiehajú matematické operácie používané v kryptografických primitívach, viď. obrázok 1.4.[12]

Neprepisovateľná pamäť ROM obsahuje jednoduchý operačný systém na ovládanie mikroprocesora, ktorý je vložený pri výrobe tagu. Operačný systém tiež obsluhuje prácu s pamäťou typu EEPROM. Na krátkodobé ukladanie slúži integrovaná pamäť RAM, ktorá je volatilná.

Ich funkčný program, alebo aplikáciu je možné nahrať až po výrobe a meniť ju. Multiaplikačné tagy dokážu obsuhovať viacero nezávislých aplikácií, ktoré buď môžu zdieľať spoločnú časť pamäte, pre ich jednoduchšiu spoluprácu, alebo môžu byť oddelené v samostatnom kontajneri. Oddelenú pamäť využívajú hlavne aplikácie, pri ktorých záleží na bezpečnosti a nedosatiteľnosti, ako e-peňaženka a podobne. [13]

V súčasnej dobe je vytvorených viacero druhov programovateľných čipových kariet, pričom najčastejšími sú Java Card pre programovanie v Jave, MULTOS - karta

podporujúca jazyk C, Basic Card - pre programovanie v jazyku Basic a .NET Card. Všetky spomínané obsahujú vlastný špecifický operačný systém.



Obr. 1.4: Mikroprocesorový tag

JavaCard

JavaCard má v sebe obsiahnutý JCRE(Java Card Runtime Environment), podobne ako klasická Java, ktorá beži nad JRE (Java Runtime Enviroment) a dovoľuje jej byť muliplatformnou. JCRE sa skladá z operačného systému JavaCard, JCVM (JavaCard Virtual Machine) a z API ponúkajúcich prostriedky pre budovanie aplikácií.

JCVM sa skladá z dvoch častí. Jedna sa nachádza priamo na čipovej karte, tá interpretuje bajtkód, spravuje triedy a objekty. Druhá časť je externá inak nazývaná ako JavaCard Converter tool, ktorá sa stará o načítanie, overovanie a prípravu kódu na spustenie na čipovej karte.

JCVM podporuje iba obmedzené množstvo z originálneho programovacieho jazyka Java. Zachováva však jej stavebné kamene ako objekty, rozhrania, výnimky či dedičnosť.[36]

Všetky vyššie spomenuté prvky sú súčasťou SDK, ktoré je voľne stiahnuteľné.[32] Pre JavaCard verzie 2.2.2 je nutné použiť SDK verzie 2.2.2 atď.

MULTOS

MULTOS (Multiple Operating System) je operačný systém, ktorý dovoľuje viacerým aplikáciám byť uloženým a bežať na jedinej čipovej karte. Každá aplikácia je separovaná firewallom tak, aby nemohla ovplyvniť inú a zároveň aby bola sama neovplyvniteľná.

MULTOS sa skladá z dvoch unikátnych technológií. Prvá je virtuálny stroj, ktorá spúšťa nahrané applety a teda sa nachádza priamo na čipe a tzv. operačná schéma MULTOS, ktorá ochraňuje mikroprocesor, kód a dáta aplikácií. Umožňuje aby boli aplikácie nahrávané bezpečne aj v kritickom prostredí. Zabezpečuje to unikátny RSA kľúč, ktorý je obsiahnutý v každom zariadení MULTOS, generovaný certifikačnou

autoritou. Pomocou verejného kľúča je pripravený software určený pre danú čipovú kartu, tzv. ALU (Application Load Unit). Iba cieľové zariadenie dokáže rozšifrovať konkrétne ALU.

Každá čipová karta MULTOS obsahuje sadu podporovaných inštrukcií, pričom môže obsahovať aj doplnkové inštrukcie, ktoré sa líšia v závislosti od výrobcu a verzie operačného systému.

1.2.4 Rozdelenie podľa spôsobu napájania

Pasívne tagy

Pasívne tagy nemajú vlastný zdroj napätia. Energiu na moduláciu vĺn príchodzích od čítačky indukujú z vĺn samotných. Anténa implementovaná v tagu reaguje na elektromagnetické pole, ktoré čítačka vytvorí a to poskytuje energiu na činnosť tagu, teda aj prenesenie informácií od čítačky ku tagu a naopak. Tag je preto spôsobilý operovať iba vrámci dosahu čítačky, ak je alokovaný mimo neho, je bez zdroja napätia a teda neschopný svoje dáta vysielať, tým pádom je jeho operatívny dosah menší.[6]

Aktívne tagy

Aktívne tagy nepotrebujú indukciu ako zdroj prúdu pre svoju funkčnosť, pretože obsahujú vlastný zdroj elektrickej energie. Zvyčajne ide o miniatúrnu baterku. Keďže sa aktívny tag nemusí obmedzovať v spotrebe energie ako pasívne alebo semipasívne tagy, je možné ho vybaviť nadštandardným hardvérom. Podľa jeho špecializovaného určenia môže byť vybavený rôznymi senzormi a vstupnými a výstupnými zariadeniami, čo však zároveň zvyšuje jeho výrobnú cenu, veľkosť a spotrebu.

Existujú dva typy aktívnych tagov. Prvým je tzv. „transponder“, ktorý komunikuje s čítačkou iba po k nemu nasmerovanej výzve, druhým je „beacon“, ktorý vysiela konštantne. Z dôvodu, aby si nemusel namáhavo vyjednávať časovanie vysielania vysiela v tzv, bitových pulzoch, teda prerušovane.

K výhodám aktívnych tagov patrí schopnosť komunikácie s čítačkou v rádoch desiatkách až stovkách metrov.[14]

Semi-pasívne tagy

Semi-pasívne tagy fungujú na podobnom princípe ako tagy pasívne, teda mechanizmus vysielania informácie je postavený na modulácii vĺn príchodzích od čítačky. Navyše však obsahujú batériu, ktorá však na rozdiel od aktívnych tagov napája iba ich hardvér. Tým pádom môžu obsahovať mikročipy či senzory a zároveň všetku energiu prijatú od čítačky využiť na moduláciu vĺn pri posielaní odpovede. Maximálna vzdialenosť komunikácie je teda väčšia ako pri tagoch pasívnych.[15]

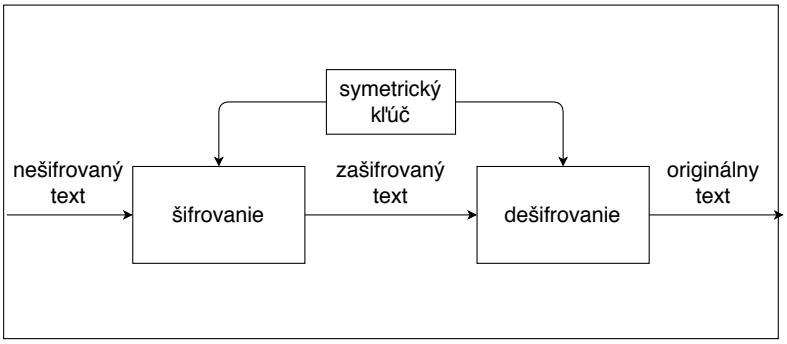
1.3 Kryptografia na RFID

Hlavnou funkciou RFID technológie je identifikácia a autentizácia užívateľov, zvierat a vecí. Najjednoduchšie systémy naozaj len identifikujú a tagy iba vysielajú svoje produktové číslo čítačke. Týmto spôsobom však nikdy nemôže byť naozaj potvrdené, či identifikácia nebola podvrhnutá. Tomuto zamedzujú autentizačné protokoly. Aj keď je výpočetná sila tagov minimálna, existuje možnosť implementovať do nich kryptografické primitíva pomerne dostačujúcej spoľahlivosti.

Cieľom kryptografie je utajiť informáciu tak, aby bola zrozumiteľná len po aplikovaní špeciálnej znalosti. Základným delením kryptografie je delenie na kryptografiu symetrickú a asymetrickú.

1.3.1 Symetrická kryptografia

Symetrická kryptografia využíva k zašifrovaniu správy rovnaký, alebo triviálne odvoditeľný kľúč ako na dešifrovanie, viď. obrázok 1.5. Tento kľúč sa teda stáva spoločným tajomstvom a jeho zverejnenie šifrovanie znehodnotí. Symetrická kryptografia sa bližšie rozdeľuje na prúdovú a blokovú. [16]



Obr. 1.5: Kryptografia za použitia symetrického kľúča

Prúdová šifra

Pri prúdovej šifre sú postupne jednotlivo bity alebo bajty šifrované symetrickým kľúčom. Pre zvýšené zabezpečenie býva kľúč pseudonáhodne generovaný pri každom jednom šifrovaní. To chráni šifru pred napadnutím hrubou silou, teda systematickým skúšaním možných kľúčov. Nevýhodou je náročná prenositeľnosť samotného kľúča, ktorý sa musí prenášať cez zabezpečený kanál a takisto aj nutnosť generovať kľúč rovnako dlhý alebo dlhší ako je daná správa. Výhodou oproti blokovej šifre je napríklad malá prejaviteľnosť chyby, kedy chyba komunikačného kanálu sa prejaví iba v jednom zodpovedajúcom znaku otvoreného textu, pričom u šifry blokovej chyba

ovplyvní celý blok. Najčastejšie je využívaná pri šifrovaní a dešifrovaní v reálnom čase. Nevýhody vo väčšine prípadov prevažujú a preto sa v realite častejšie vyskytujú šifry blokové.[17]

V RFID systémoch je prúdová šifra rozšírená v čipových kartách MIFARE Classic. Tieto karty používajú propietárnu šifru CRYPTO1, ktorá je postavená práve na prúdových základoch.[18]

Bloková šifra

Bloková šifra, ako už napovedá názov, pracuje s bitmi v blokoch, kedy v jednom bloku je n-bitov.V niektorých typoch blokových šifier platí, že ak sa správa nedá presne zadeliť do blokov, teda jej počet bitov nie je deliteľný n bezo zvyšku, musí sa daný blok doplniť, ide o tzv. padding.

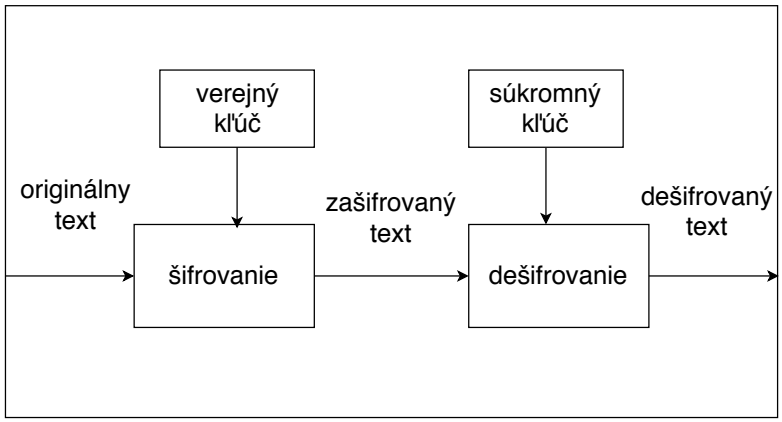
Vzájomné vzťahy medzi jednotlivými blokmi kryptogramu definujú módy blokových šifier. Najzákladnejší je mód ECB (Electronic Codebook), kde je každý blok samostatne zašifrovaný symetrickým kľúčom. Próblémom je, že daný vstupný blok sa vždy transformuje na zašifrový blok rovnakej podoby a teda má nízku difúziu. Aby sa zabránilo tomuto stavu, pridávajú sa na začiatok šifrovania náhodné inicializačné vektory a šifrovanie sa reťazí, čiže výstup šifrovania daného bloku závisí na bloku predchádzajúcom. To popisuje napríklad mód CBC (Cypher Block Chaining), ktorý je v dnešnej dobe najpoužívanejším módom blokovej šifry.[19]

Blokové šifrovanie využívajú napríklad čipové karty MIFARE DESFire, ktoré sú založené na šifre 3DES alebo AES. DES pracuje s s blokmi o dĺžke 64 bitov a cez radu matematických operácií blok zašifruje. Dĺžka kľúča je v prípade DES 64 bitov, pričom 56 bitov je pracovných 8 bitov kontrolných. 3DES prevedie algoritmus DES 3x za sebou, pričom využíva kľúč o dĺžke 168 bitov, čo zvyšuje jeho bezpečnosť. AES šifruje bloky o veľkosti 128 bitov a využíva kľúč o dĺžke 128, 192 alebo 256 bitov. Je rýchlejší a bezpečnejší ako 3DES.[18]

1.3.2 Asymetrická kryptografia

Na dešifrovanie sa používa kľúč súkromný, ktorý je bez jeho znalosti neodvoditeľný od kľúča verejného, ktorým sa dáta šifrujú. Opačným smerom však existuje jednoduchá funkcia pomocou ktorej sa verejný kľúč odvodí od súkromného. Tá zabezpečí, aby spolu kľúče matematicky súviseli. Kľúč na šifrovanie môže byť ľubovoľne rozdistribuovaný, keďže neohrozí tajomstvo dát. Princíp asymetrického šifrovania je graficky znázornený na obrázku 1.6.

Medzi matematické problémy na ktorých sú založené súčasné asymetrické protokoly patrí napríklad problém faktorizácie, problém diskrétneho logaritmu a problém diskrétneho logaritmu nad eliptickými krivkami.



Obr. 1.6: Kryptografia za použitia asymetrického kľúča

Problém faktorizácie

Problém faktorizácie je založený na princípe obtiažnosti rozkladu veľkého čísla na prvočísla oproti pomerne menej náročného vzniku veľkého čísla vynásobením prvočísel. Rozklad je vždy deterministický, teda spätným procesom sa vždy príde k prvotnému číslu. Kľúče generuje strana prijímateľa a verejný kľuč rozdistribuuje.[20]

Problém diskrétneho logaritmu

Problém diskrétneho logritmu podobne ako problém faktorizácie popisuje v podstate jednosmernú funkciu, kedy je jej opačný smer nemožné so súčastnými technickými prostriedkami vypočítať v reálnom čase. Ak poznáme x a g, pričom platí, že $x, g \in Z_p^*$ a poznáme tiež veľké prvočíslo p, potom $y \equiv g^x \pmod{p}$ je spočítateľné jednoducho, avšak opačný proces, teda zistenie mocniny x so znalosťou y, g a p je výpočetne nemožné. Preto sa parameter x stáva tajomstvom, na ktorom je postavená bezpečnosť šifrovania.[16]

Eliptické krivky

Rovnicu eliptickej krivky je možné zapísať ako $y^2 = x^3 + ax + b$. Ak je postavená nad konečným polom F_p, tak $a, b \in F_p$.

Ak máme body eliptickej krivky P a Q a celé číslo d je pomerne jednoduché vypočítať $Q = dP$, ale komplikované vypočítať d pri znalosti P, Q, teda ide o matematický problém diskrétneho logaritmu na eliptickej krivke. Preto sa z d stáva súkromný kľúč a z $Q = dP$ kľúč verejný.

Výhodou eliptických kriviek je vo všeobecnosti kratšia nutná dĺžka kľúča pri udržaní podobnej bezpečnosti, ako pri použití kryptografických prostriedkov na báze

problému faktorizácie. To je umožnené vďaka väčšej matematickej zložitosti systému eliptických kriviek. [21]

1.4 Súkromie a bezpečnosť

Pri bezpečnosti RFID technológií sa zavádzajú najmä pojmy identifikácia, autentizácia a riadenie prítupu.

Identifikácia

Identifikácia je schopnosť zistiť užívateľovu unikátnu identitu. Inými slovami, je to zistenie totožnosti daného užívateľa.

Autentizácia

Autentizácia je proces overenia identifikácie daného užívateľa. Ide o bezpečnostný postup zabraňujúci falzifikácii osobnosti. Užívateľ sa môže autentizovať napríklad informáciou - vie to, čo nikto iný nevie a preukázaním tejto informácie potvrdí svoju identitu, alebo jeho jedinečnou vlastnosťou, teda využitie aplikácie metód biometrie. Takto sa dokáže autentizovať hlasom, očnou sietnicou či odtlačkom prstu. [23]

Kontrola prístupu

Kontrola prístupu je ochrana pred vstupom nepovolaných osôb do daných objektov, či priestorov pomocou tzv. prístupových systémov. Pomocou kontroly prístupu je takisto možné užívateľovi bližšie špecifikovať práva podľa povahy jeho práce a na ich základe mu dovoliť vykonávať určité činnosti. [23]

Bezpečnosť RFID technológií

S rozmachom RFID identifikačného systému sa vynára otázka súkromia a bezpečnosti. Tag odpovedá na každú výzvu kompatibilnej čítačky, a to i bez povedomia majiteľa. Aj keď privátne a tajné informácie je už možno bezpečne zašifrovať aj na takých obmedzených zariadeniach ako sú RFID tagy, stále ostáva faktom, že UID tagu, aj keď môže byť náhodne generované a na prvý pohľad nenesie žiadnu dôležitú informáciu, je voľne prístupné v čitateľnej forme.

Väčším rizikom sú multiaplikačné tagy, na ktorých zároveň koexistujú aplikácie pre viaceré použitia. Typicky to môže byť karta do knižnice, či lístok na mestskú hromadnú dopravu. Chovanie jednotlivca by nemalo byť vystopovateľné, zreprodukovateľné a jednoznačne pripraditeľné k jeho osobnosti. Riziko bude s väčším využívaním RFID systémov rásť.[24]

Ďalším potencionálnym problémom je klonovanie tagov. Najjednoduchšie tagy, ktoré majú svoju identifikáciu založenú len na voľne čitateľnom UID sú najjednoduchším terčom. Na špeciálne tagy bez výrobcom stanoveného UID ide pomocou zapisovacej čítačky číslo iného tagu skopírovať a tým ho podvrhnúť.

Tomuto zabraňujú autentizačné systémy v tagoch implementované. Najjednoduchšie z nich, napríklad na spomínanej karte MIFARE Classic však už boli prelomené a tým pádom je v ohrození milióny užívateľov, u ktorých sú karty stále vo využití. Mnohokrát aj keď je autentizácia dostupná býva preskakovaná a pre prístup je využité iba ID karty. U štandardných systémov je ID vždy známe, môže slúžiť k sledovaniu ľudí, ich profilovaniu a prípadne aj straty identity užívateľa. [25]

2 Praktická časť študentskej práce

Hlavná časť praktickej časti je implementácia a popis prístupového systému zloženého z RFID štítkov a čipových kariet za pomoci nenáročných počítačov.

Za týmto účelom tiež praktická časť popisuje vykonanie analýzy a výkonových testov na súčasných RFID technológiách. Na základe meraní je vybraný vhodný hardware pre autentizačný systém.

2.1 Pridelený hardvér a merania

Táto sekcia obsahuje porovnanie RFID tagov pre určenie vhodnosti využitia v ďalšej práci. Merania boli uskutočnené s pridelenými čítačkami a tagmi, s ohľadom na charakteristiky danej oblasti RFID systémov. Je nutné rozdieľne pristupovať k LF systémom a inak k systémom HF a UHF. V meraní za vyskytla aj nezabrániteľná chybovosť merania, ktorá bola spôsobená vplyvom prostredia, nespoľahlivosťou čítačiek a takisto aj ľudským faktorom.

Hlavnými bodmi merania boli **dosah čítačiek, rýchlosť identifikácie a jej chybovosť**. Nie všetky parametre sú spoľahlivo zmerateľné na všetkých čítačkách, preto boli pri niektorých vynechané.

Na testovanie účely a komunikáciu s čítačkami bol používaný operačný systém GNU/ Linux 4.18.0-2-amd64, distribúcia Debian, bežiaci na počítači s procesorom Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz a 8 GB operačnej pamäti DDR3.

2.1.1 LF systém

Pridelená RFID čítačka pracujúca na frekvencii 125 kHz je vyrobená spoločnosťou AuthenTec Inc. Je definovaná ako rozhranie človek-stroj, typu klávesnica. To znamená, že po priložení tagu jeho ID načíta ako klasický klávesový vstup a zapisovanie na RFID tagy pomocou nej nie je možné.

Jediný spoľahlivo zmerateľný parameter je dosah čítačky, ktorý je v klasickom prípade merateľný v jednotkách centimetrov. Meranie bolo aplikované na dva rôzne RFID LF tagy, prvý je NYLON 4200EM, v tvare hodiniek a druhý je taktiež typu 4002EM, ale je v tvare občianskeho preukazu. Výsledky ukázali, že existuje zlomová vzdialenosť, približne 6 cm pre hodinky a 7 cm pre plastovú kartu, po ktorej je úspešnosť načítania nulová,

2.1.2 HF systém

HF systémy sú zložitejšie, teda aj čítačka komunikujúca na HF musí ovládať viacej funkcií. Pre účely tejto práce bola poskytnutá konktrétne čítačka typu ACR122U

od spoločnosti Advanced Card Systems Ltd. S počítačom komunikuje cez USB rozhranie. Pre jej funkčnosť v systéme Linux bolo potrebné odstrániť obsiahnuté NFC moduly a nainštalovať démona *pcscd*, teda proces pracujúci na pozadí, zaisťujúci komunikáciu medzi systémom Linux a čítačkou, balík *pcsc-tools*, ktorý poskytuje základné utility na ovládanie čítačky a NFC knižnica *libpcsclite1* kvôli základom pre *pcsc-tools*, viď. výpis 2.1. [26].

Výpis 2.1: Postup prípravy prostredia pre komunikáciu s HF čítačkou

```
sudo apt install pcscd pcsc-tools libpcsclite1
sudo modprobe -r pn533_usb pn533 nfc
sudo systemctl restart pcscd
```

Z balíka *pcsc-tools* bol konkrétne použitý program `scriptor`, ktorý po jeho spustení posiela čítačke bajty príkazov zadané používateľom.

Moduly `pn533_usb, pn533, nfc` sú natívne moduly Linuxového jadra, ktoré zaisťujú NFC komunikáciu. Ak sú tieto moduly k jadru pripojené, `pcscd` reportuje chybu „Can't claim interface“, čiže je neschopný používať NFC rozhranie. Knižnica ovládajúca NFC rozhranie je však na funkčnosť čítačky nevyhnutná. Preto bola dodatočne stiahnutá a nainštalovaná knižnica `libpcsclite1`, s ktorou čítačka fungovala.

Merania

Pre HF boli na meranie využité tagy technológie MIFARE Ultralight a v poslednom meraní MIFARE DESFire EV1. Pri výbere tagov bol braný na zreteľ spôsob ich spracovania, ich tvar a rozmery antény, viď. tab. 2.1.

Tab. 2.1: Tagy použité na meranie HF RFID systému

tag	**rozmery antény**	**spracovanie**	**označenie**	**čas odozvy**
A_1	7 x 4 cm	plastová karta	MIFARE Ultralight	25,909 ms
A_2	7 x 4 cm	plastová karta	MIFARE DESFire	-
B	7 x 4 cm	nálepka	CR80PET	26,837 ms
C	3 x 3 cm	nálepka	NTAG 203	17,676 ms
D	2 x 2 cm	kruhový čip	NTAG 213	28,562 ms
E	2 x 2 cm	kľúčenka	NTAG 6213	29,318 ms
F	4 x 2 cm	náramok	NTAG 203	20,969 ms

Maximálny dosah

Maximálny dosah HF čítačky je pomerne malý, je závislý na veľkosti antény tagu, s ktorým sa práve pokúša komunikovať. Všetky tagy mali podobný dosah, teda rozdiely boli zanedbateľné. Priemerný dosah bol 5 cm.

Rýchlosť identifikácie

Meraný bol čas od výzvy počítača na vyslanie APDU príkazu po obdržanie odozvy tagu od čítačky. Príkaz APDU požadoval od tagu vrátenie jeho UID. Priemerný čas odozvy, a teda dentifikácie každého tagu je vidieť v tab. 2.1.

Meranie bolo kvôli získaniu väčšieho množstva dát zautomatizované skriptom napísaným v skriptovacom jazyku `bash`. V skripte bol použitý vyššie spomenutý program `scriptor`, ktorý periodicky raz za dve sekundy posielal čítačke APDU správu na výzvu tagu na identifikáciu. Toto bolo prevedené v cykle 100x. Pred poslaním bajtov a tesne po prijatí odpovede bol zaznamenaný aktuálny čas, ktorý sa od seba odčítal, čím bol zistený presný čas identifikácie.

Rýchlosť autentizácie na čipovej karte MIFARE DESFire

Pre porovnanie bola zmeraná aj rýchlosť dnes stále pomerne bezpečnej autentizácie na čipovej karte MIFARE DESFire EV1. Základom používanej autentizácie bola šifra 3DES, ktorá v trojcestnej výmene autentizovala kartu aj čítačku. Čas bol meraný od inicializačnej výzvy o autentizáciu vyslanej z počítača po prijatie poslednej správy trojcestnej výmeny od karty. Autentizácia bola meraná programom napísaným v jazyku `Java`, na základne `javax.smartcardio`[27] knižnice, ktorá poskytuje základ pre komunikáciu s čipovými kartami. Kroky autentizácie sú následovné:

1. Počítač vyzve kartu skrz čítačku na prvotnú autentizáciu. Pri použití šifry 3DES, ako v tomto prípade, je použiý APDU príkaz `"1a 00"`.
2. Karta pošle späť náhodne vygenerovné pole 8 bajtov, ktoré je zašifrované jej hlavným kľúčom k.
3. Počítač spracuje prijaté bajty, dešifruje ich šifrou 3DES s použitím kľuča k a urobí na nich bajtový posun, čím z nich vznikne `randB`.
4. Počítač vygeneruje náhodné pole o dĺžke 8 bajtov(`randA`). Za `randA` pripojí `randB`. Výsledných 16 bajtov zašifruje a pošle karte.
5. Karta prijaté bajty dešifruje, odseparuje z nich `randA`, bajtovo ho posunie o jeden bajt, zašifruje a pošle počítaču.
6. Počítač správu odšifruje a bajtovo posunie späť. Výsledok porovná s pôvodným `randA`.
7. Ak sú identické autentizácia prebehla úspešne. Z `randA` a `randB` sa može zložiť konkrétny `session key`, teda kľúč používaný vrámci celej komunikácie do odpojenia karty.

Postup autentizácie je badateľný vo výpise z `Java` programu, viď. výpis 2.2.

Meranie prebehlo 10x, výsledky boli spriemerované. Výsledný čas autentizácie bol 144.52 ms.

Výpis 2.2: Výpis z autentizácie čipovej karty MIFARE DESFire

```
Terminals: [PC/SC terminal ACS ACR122U 00 00]
CARD:  PC/SC card in ACS ACR122U 00 00, protocol T=1, state
   OK
UID:     = 0x04 53 06 A2 13 4E 80  len(7)
RESP1:   = 0x8C 00 5B 51 44 23 CC 11 91 AF  len(10)
ENC_B:   = 0x8C 00 5B 51 44 23 CC 11  len(8)
RAND_B:  = 0x52 01 2E C2 26 58 16 21  len(8)
B_ROT:   = 0x01 2E C2 26 58 16 21 52  len(8)
RAND_A:  = 0xC0 75 19 CB D3 44 22 37  len(8)
RAND_AB: = 0xC0 75 19 CB D3 44 22 37 01 2E C2 26 58 16 21 52
    len(16)
ENC_AB:  = 0xA1 CC E7 8F B9 5D 11 3D E5 3C 4E 8C 5C EA C4 8C
    len(16)
RESP2:   = 0x42 53 79 D5 85 FC F2 0B 91 00  len(10)
RESP_A:  = 0x42 53 79 D5 85 FC F2 0B  len(8)
ENC_IV:  = 0xE5 3C 4E 8C 5C EA C4 8C  len(8)
A_ROT:   = 0x75 19 CB D3 44 22 37 C0  len(8)
RAND_A:  = 0xC0 75 19 CB D3 44 22 37  len(8)
SESSION KEY:
 = 0xC0 75 19 CB 52 01 2E C2 D3 44 22 37 26 58 16 21  len(16)
EXECUTION TIME: 137.984434ms
```

2.1.3 UHF systém

Poskytnutá UHF čítačka je typu UHFreader18. Dokáže interagovať s tagmi spadajúcimi pod štandard ISO18000-6B a ISO18000-6C. Spojenie s počítačom zaisťuje sériová linka RS232. K čítačke bola dodaná knižnica `UHFReader18CSharp.dll` napísaná v jazyku C# a GUI (Graphical User Interface) nadstavba `UHFReader18demomain.exe` pre komunikáciu s operačným systémom Windows, datasheet a manuál na softvér komunikujúci z čítačkou. Manuál uvádza komunikáciu čítačky na frekvenciách 902 - 928 MHz, jej výstupná sila môže byť nastavená až do výšky 30 dBm. Zisk antény je 8 alebo 12 dBi.[28]

Čítačka je ovládaná z počítača vždy sledom bajtov, viď. tab. 2.2. Výzva môže byť typu `set` (nastavuje), `get` (žiada výpis nastavenia) alebo typu žiadosť o akciu. Všetky typy majú rovnakú formu, u žiadosti o výzvu však nie sú obsiahnuté žiadne dáta. Odpoveď je rovnakej formy ako výzva, s pridaným bajtom pre návratový kód.

Čítačka si po pripojení dohodne svoju adresu s počítačom, pričom, sa začína od `00`. Táto adresa sa využíva v nasledujúcej komunikácií, je zapísaná v bajtovom poli `addr`. Funkcia je určená pre možnosť pripojenia viacerých čítačiek súčasne. Za účelom všesmerového adresovania sa použije adresa `ff`.[28]

Pre využitie čítačky na platforme Linux bolo potrebné použiť externú knižnicu UHFReader18[29], ktorá s týmto sledom bajtov pracuje a implementuje funkcie potrebné na komunikáciu s čítačkou, ako pridanie kontrolného súčtu CRC-16, adresovanie komunikácie a správu konštánt pre jednoduchšie vytváranie nadstavbových programov. V knižnici, však nebola obsiahnutá funkcia na načítanie ID tagu, teda musela byť osobitne implementovaná, viď. výpis 2.3. Čítačke je pomocou metódy `send` obsiahnutej v knižnici UHFReader18 na broadcastovú adresu vyslaný príkaz `query_tag`, ktorý je určený číslom `0x01` v poli `cmd`. Funkcia vracia okrem návratových kódov čítačky aj ID tagu, ak je toto zosnímané čítačkou. Samotná čítačka je schopná tagy čítať, aj na ne zapisovať.

Výpis 2.3: Funkcia tagQuery(), dodatočne pridaná do modulu UHFReader18

```
#self.QUERY_TAG = 0x01                                                    1
def tagQuery(self):                                                       2
     self.send(self.ADDR_BROADCAST, self.QUERY_TAG)                       3
     recv = self.recv()                                                   4
     return (recv[0], recv[2], recv[5:-2] if recv[2] == 0x01              5
        else None)
```

Tab. 2.2: Organizovaný sled bajtov posielaný z počítača čítačke na jej ovládanie

poradie	**názov**	**funkcia**	**počet bajtov**
1	`len`	počet bajtov celého sledu bez bajtu `len`	1
2	`addr`	adresa čítačky	1
3	`cmd`	konkrétny príkaz	1
2	`ret_val`	návratový kód (iba v odpovedi)	1
4	`data`	dáta	`len` - 5/ `len` - 4
5	`checksum`	CRC-16	2

Merania

Čítačka bola pri základnom meraní nastavená na silu vysielania 10 dBm. Na meranie bola pridelená vzorka pasívnych UHF tagov od spoločnosti Omni-ID, pasívny

tag v tvare náramku a pasívny tag v tvare občianskeho preukazu. Vzorka tagov pozostávala zo skupín tagov rôznych vyhotovení, a to zo skupiny nálepiek, zo skupiny miniatúrnych čipov a zo skupiny väčších krabičiek. Na všetkých tagoch prebehlo základné meranie, ktoré mapovalo maximálnu vzdialenosť tagu pre komunikáciu s čítačkou.

Po vyhodnotení merania bol stanovený záver, že k dôkladnému meraniu z každej skupiny vyhotovenia prejdú tie tagy, ktré sa navzájom odlišujú funkčnými schopnosťami, nebudú sa teda merať dva tagy s podobným vyhotovením a zároveveň s podobnými RFID vlastnosťami, teda v podstate s podobnou dĺžkou dosahu. V skupine nálepiek sa tagy chovali natoľko podobne, že postačujúcim bolo vybrať jedného zástupcu, konkrétne tag IQ 100. V skupine čipov a v skupine krabičiek sa tagy vyčlenili na dve funkcionálne podskupiny, a teda z každej bol k meraniu pripustený jeden. Z čipov to boli FIT 200 a FIT 210 a z krabičiek EXO 400 a EXO 800P. Ďalej sa k meraniu vyhradil pasívny tag v tvare náramku a pasívny tag - plastová karta, viď tab. 2.3. Ak sa tagy v skupine podobali RFID vlastnosťami výber prešiel na parameter spracovania, pričom preferované boli tagy v menšom prevedení, ktoré budú praktickejšie k ďalšiemu využitiu.

Finálne bolo k meraniu vybratých 7 rozdielnych UHF RFID tagov, ktoré sa odlišovali napájaním, veľkosťou antény a formou spracovania a maximálou možnou komunikačnou vzdialenosťou ku čítačke, viď. tab. 2.3. Meranie tagov mohlo do istej miery skresliť využité interiérové meracie prostredie. Pri meraní sa tagy nachádzali v priamom dohľade od čítačky, umiestnené priamo oproti nej. Medzi tagmi a čítačkou neboli žiadne prekážky, merania prebiehali na vnútornej chodbe o šírke asi 2,5 m. Čítačka sa pri meraní nachádzala asi 0,7 m nad zemou.

Tab. 2.3: Tagy použité na meranie UHF RFID systému

zn.	**typ napájania**	**rozmery antény**	**názov**	**forma spracovania**
A	pasívne	7 x 4 cm	KARTA	plastová karta
B	pasívne	2,5 x 2,5 cm	IQ 100	nálepka
C	pasívne	0,5 x 0,5 cm	FIT 200	čip
D	pasívne	2 x 1 cm	NARAMOK	náramok
E	pasívne	6 x 3 cm	EXO 800P	plátok
F	pasívne	1,5 x 0,5 cm	EXO 400	krabička
G	pasívne	4 x 0.4 cm	FIT 210	čip

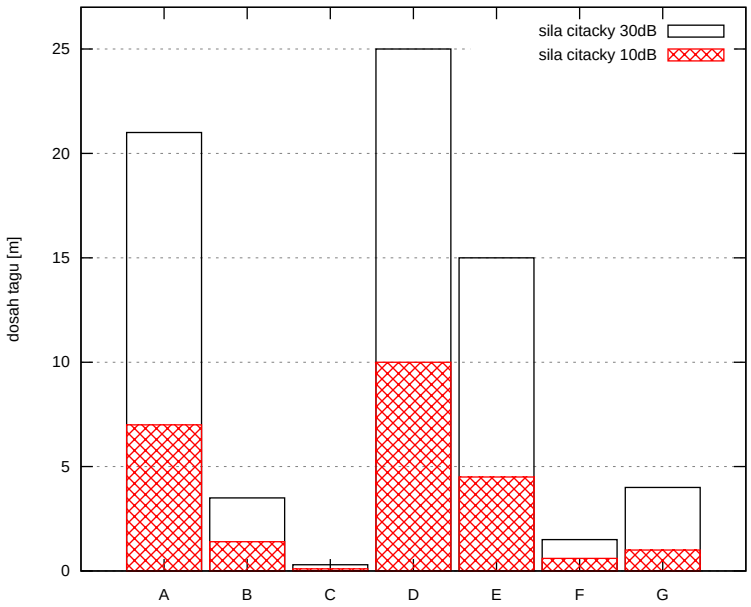
Maximálny dosah

Pri maximálnej sile čítačky bolo testovaných daných 7 referenčných tagov na najväčš-

siu možnú vzdialosť pre úspešnú komunikáciu s čítačkou. Príjem tagov silne závisel od prostredia, no i tak boli poznať značné rozdiely medzi konkrétnymi tagmi. Tagy s väčšou veľkosťou antény mali vo všeobecnosti väčší dosah, ako tagy s anténou menšou. Najväčší dosah mal tag v tavre náramku, viď. obrázok 2.1, ktorý dosiahol na vzdialenosť 25 m, ďalej plastová karta s dosahom 21 m. Najmenší dosah mal čip, čo je úmerné veľkosti jeho antény. Išlo o približne 30 cm.

Maximálny dosah pri referenčnej sile čítačky

Tento parameter bol meraný podobne, ako maximálny dosah pri maximálnej sile čítačky, sila však bola na začiatku nastavená na referenčnú hodnotu. Ako referenčná sila čítačky bola zvolená sila 10 dBm. Vybraná bola kvôli spresneniu merania, keďže maximálna vzdialenosť dosahu čítačky pri jej plnej sile môže byť desiatky metrov, čo by sa viacej prejavilo na chybe merania spôsobenej prostredím.Táto hodnota bude používaná v ostatných meraniach tejto čítačky, ak nebude napísané inak. Pri tejto hodnote bol odmeraný dosah tagov, viď. obrázok 2.1, pričom najväčší dosah mal opäť tag v tvare náramku a to približne 10 m. Čip s anténou 0,5 x 0,5 cm podľa očakávania dosiahol len na veľmi krátku vzdialenosť, konkrétne 10 cm.



Obr. 2.1: Maximálny dosah každého tagu pri sile čítačky 10 dBm a 30 dBm

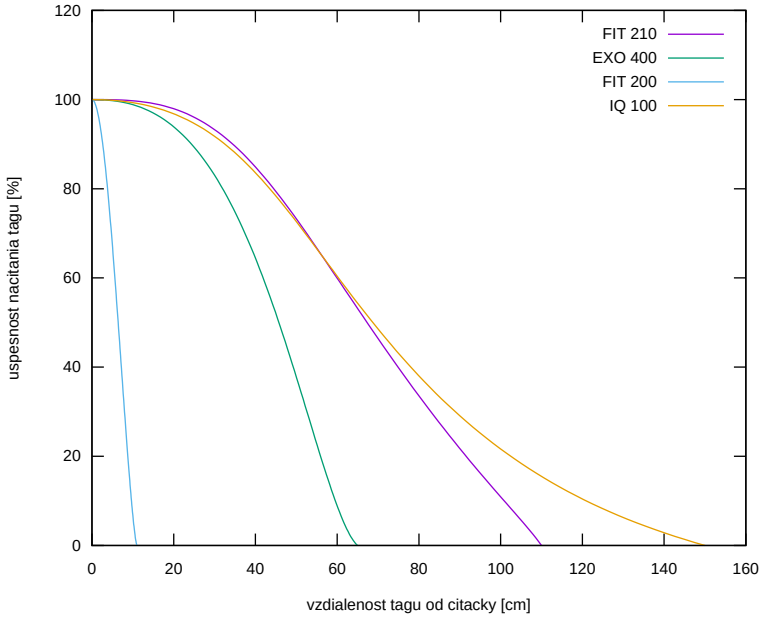
Chybovosť

Chybovosť bola meraná v závislosti na vzdialenosti tagu od čítačky. K meraniu sa použil modifikovaný `python` skript, ktorý počítal počet úspešných a neúspešných identifikácií. Z týchto hodnôt vyrátal percentuálnu úspešnosť, viď výpis 2.4. Tagy

boli merané z viacerých vzdialeností a výsledky boli pre lepšiu prehladnosť vynesené do dvoch grafov, a to do grafu 2.2 a 2.3. Merania boli vždy ukončené pri 40 úspešných pokusoch.

Výpis 2.4: Výpis skriptu na testovanie UHF systému

```
pocet korektnych identifikacii je: 100                          1
pocet vsetkych pokusov je: 253                                  2
priemerny cas identifikacie je: 37.8120589256ms                 3
percentualna uspesnost identifikacie je: 39.5256916996          4
```



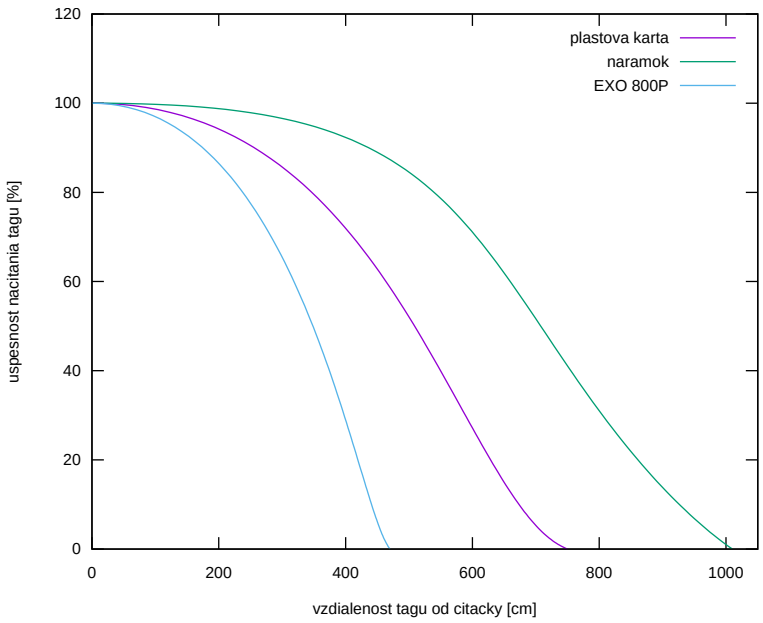
Obr. 2.2: Úspešnosť identifikácie UHF systému č. 1

Rýchlosť identifikácie

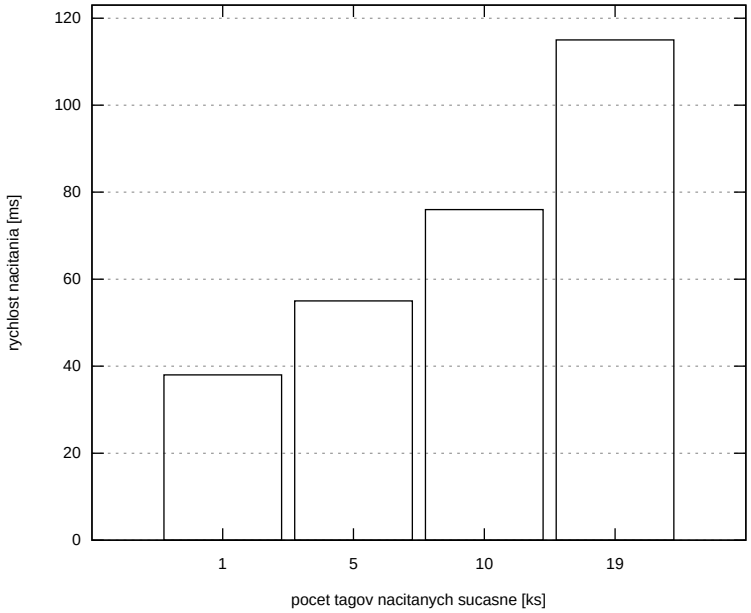
Meraný bol čas od výzvy počítača o vyslanie EPC tagu po úspešné vrátenie hodnoty od čítačky. V prvej variante merania bola vzdialenosť tagu od čítačky 0 cm. Samotný čas bol zaznamenaný funkciou `time()` z python knižnice `time`. Výsledky všetkých tagov boli zanedbateľne rozdielne, kedy meraný proces trval 38 ms.

V druhej variante bola vzdialenosť tagu od čítačky striedavo 2 a 3 m, kde sa zistilo, že čas identifikácie nie je v hmatateľnej miere závislý na vzdialenosti, ak tá stačí na nabitie tagu a na vyslanie jeho odpovede.

Merací skript nastavil čítačku do stavu, aby bola schopná tag načítať a potom v cykle posielal výzvy tagu na identifikáciu. Skript eliminoval závislosť na chybovosti vzhľadom na väčšiu vzdialenosť tagu, a to takým spôsobom, že do výsledných



Obr. 2.3: Úspešnosť identifikácie UHF systému č. 2



Obr. 2.4: Rýchlosť načítania viacerých UHF tagov zároveň

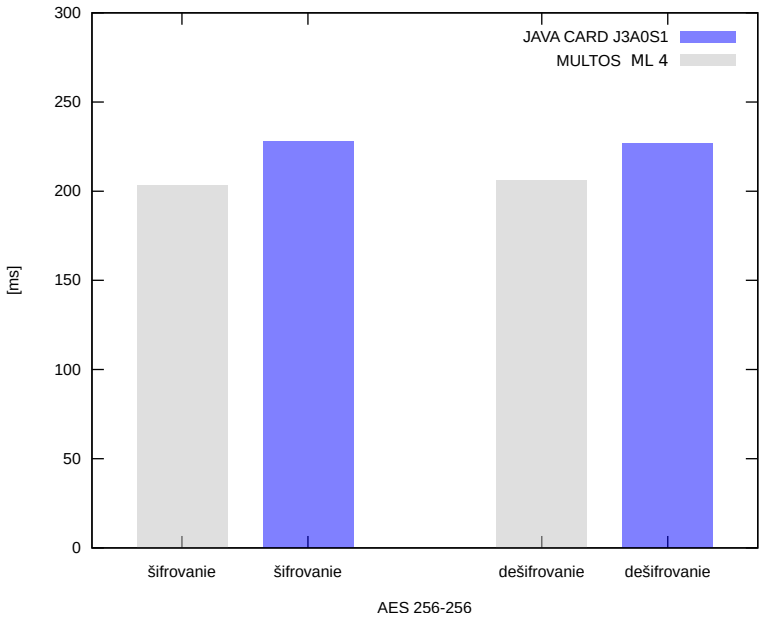
meraní sa započítaval len čas, v ktorom prebehla identifikácia úspešne. Neúspešné pokusy, ktoré môžu trvať dlhšie a meranie tak znehodnotiť boli vyradené podmienkou úspešnosti. Vzorka na spriemerovanie pozostávala vždy zo 100 identifikácií.

Meranie rýchlosti načítania EPC tagu bolo merané aj pri viacerých tagoch naraz, a to pri referenčnej vzdialenosti tagu od čítačky 1,5 m. Výsledky merania sú vykres-

lené v obrázku 2.4. Výsledný čas je vždy pre všetky tagy načítané spoločne, pretože čítačka načítané tagy zoraďuje za sebou do sledu bajtov a počítaču odošle vždy celý sled naraz. Pre rozlíšenie toho, kedy začína EPC konkrétneho tagu a kedy končí sú tieto rozdelené bajtom „`0c`“. Najviac EPC, ktoré sa podarilo čítačke preniesť v jednej odpovedi bolo 19. Ak čítačka mala vrátiť viac ako 19 EPC, vracala namiesto nich nezmyselnú odpoveď. Načítanie 19 tagov zároveň je približne 6x rýchlejší proces ako postupné načítanie 19 tagov.

2.2 Meranie kryptografických primitív

V tejto sekcii sa nachádza zhrnutie najdôležitejších kryptografických primitív dostupných na čipových kartách. Porovnaná je rýchlosť symetrickej šifry AES, asymetrického kryptosystému RSA (Rivest–Shamir–Adleman cryptosystem) a kryptosystému na bázi eliptických kriviek. Pre porovnanie ráchlostí boli vybraté čipové karty MULTOS ML4 a Java Card.



Obr. 2.5: Rýchlosť AES na čipových kartách

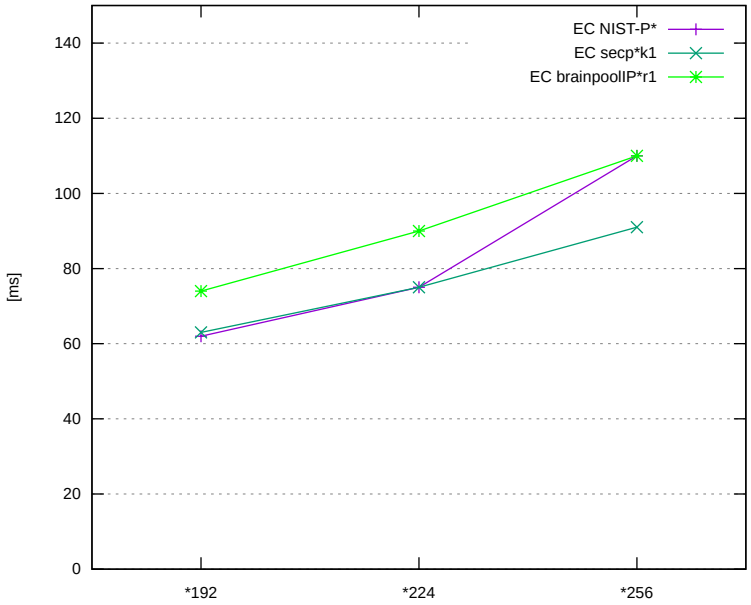
AES

AES je bloková symetrická šifra. Pre porovnanie rýchlosti bol vybraný typ, ktorý pracuje s o veľkosti 128 bitov. Tajný kľúč je o veľkosti 256 bitov. Na obrázku 2.5 je vidieť, že čipová karta MultOS vypočítala tento algoritmus rýchlejšie ako karta Java

Card. Taktiež je viditeľné, že šifrovanie i dešifrovanie pomocou AES zaberú danej karte podobné množstvo času.

Kryptografia eliptických kriviek

Ako prvý parameter porovnávania bol vybraný čas násobenia dvoch bodov na eliptickej krivke, čo je spolu so sčítaním bodov na eliptickej krivke hlavným pilierom kryptosystémov založených na eliptických krivkách. Porovnané boli eliptické krivky nad prvočíselným polom podľa štandardu NIST (National Institute of Standards and Technology), secp a brainpool o 192, 224 a 256 bitoch. Výsledky pre Java Card hardvér sú vynesené v grafe 2.6.



Obr. 2.6: Rýchlosť násobenia bodov na eliptickej krivke na čipovej karte Java Card

2.3 Autentizačný protokol

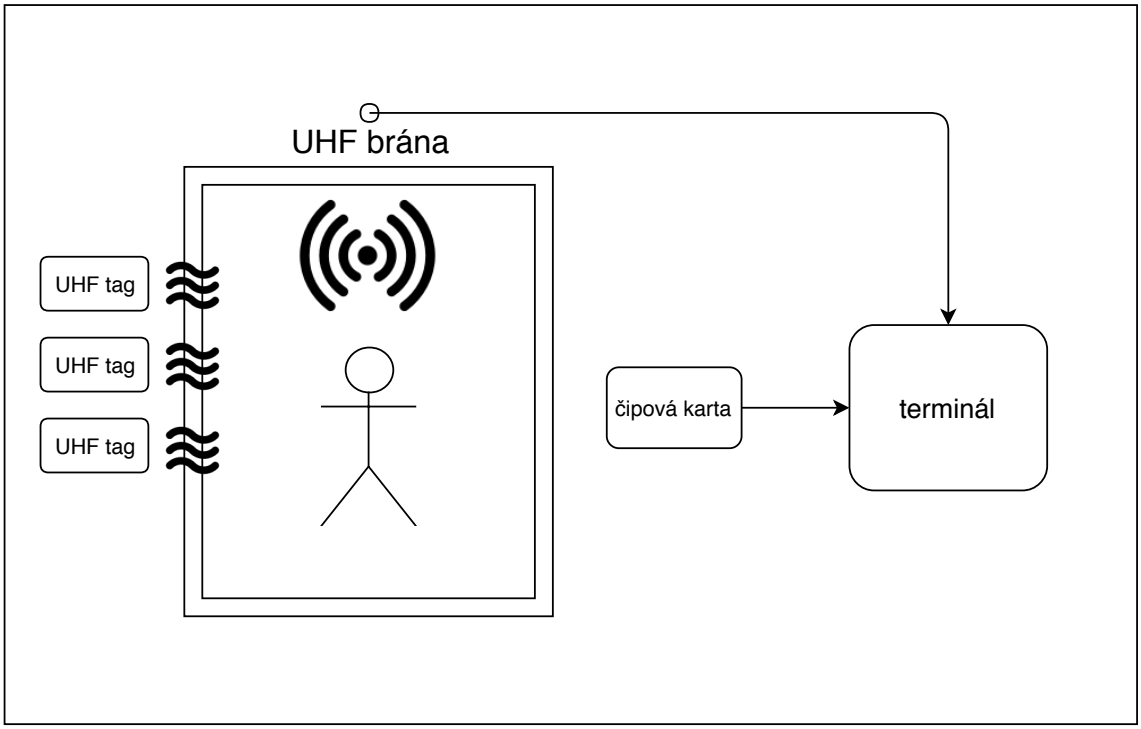
Hlavným prvkom praktickej časti bakalárskej práce je implementácia autentizačného protokolu. Stanovené požiadavky sú:

- uživateľská jednoduchosť - systém by teda mal byť bezkontaktný,
- rýchlosť - užívateľ musí byť autentizovaný za dobu, ktorá ho nebude obťažovať,
- ochrana súkromia užívateľa - na základe porovnania komunikácie by nemalo byť možné určiť identitu užívateľa,
- spoľahlivá autentizácia,
- využitie UHF štítkov vrámci viacfaktorovej autentizácie.

2.3.1 Architektúra autentizačného systému

V autentizačnom systéme sa využíva viac zariadení zároveň, a to bezkontaktná HF čipová karta s mikroprocesorom podporujúcim kryptografické operácie, ktorá je priradená užívateľovi, a viacero vedľajších, jednoduchých identifikátorov, UHF tagov, ktoré majú spomedzi RFID najväčší dosah, a dokážu byť čítačkou načítané viaceré zároveň.

Ďalším prvkom je SAM (Secure Access Module), teda výpočetné zariadenie, ktoré spracováva komunikáciu s užívateľovou čipovou kartou a drží bezpečne uložené tajné kľúče. Musí ísť o zariadenie odolné voči narušeniu (tamper-proof device). Je fyzicky pripojený k terminálu, ktorý smeruje informácie medzi SAMom a čipovou kartou. V terminále, teda v autentizačnom serveri je takisto uložená databáza registrovaných čipových kariet s ku nim priradených tagov. Pri registrácii je využitý registračný server, ktorý generuje kľúče a parametre účastníkom protokolu. Schéma autentizačnej časti systému je znázornená na obrázku 2.7.

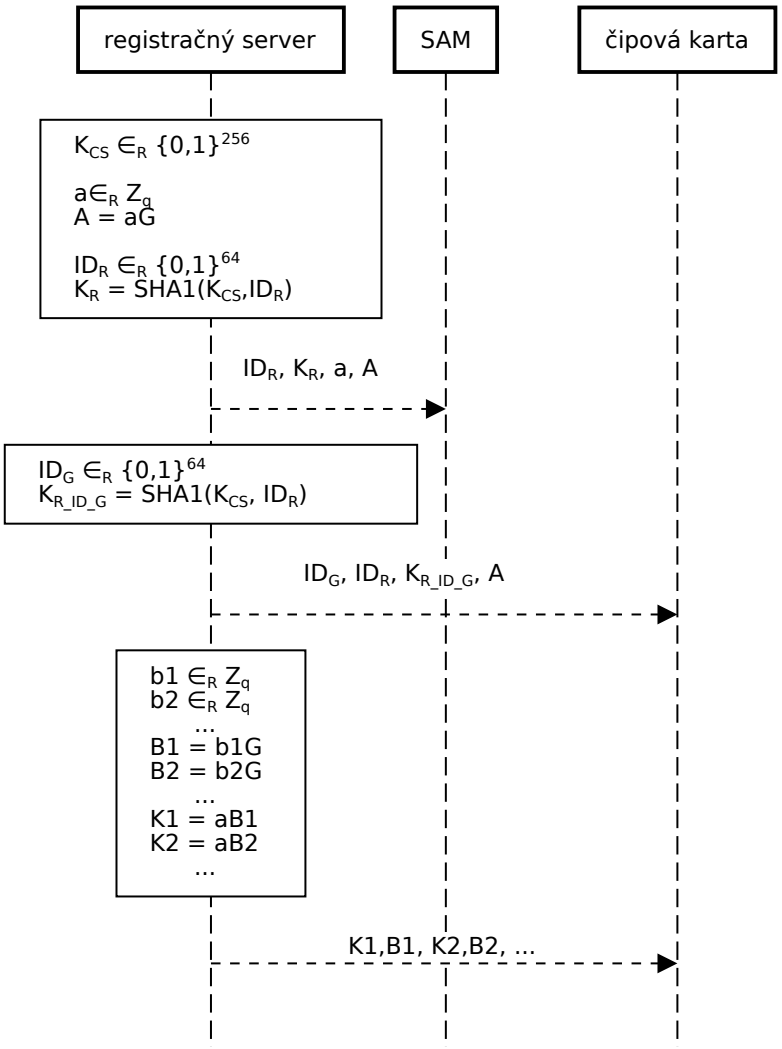


Obr. 2.7: Navrhovaný scénar autentizačného systému

Systém je navrhnutý ako automatická detekcia ochranného vybavenia užívateľa, ktorý sa autentizuje čipovou kartou. Overenie užívateľa však nebude úspešné bez doplnkového identifikovania sa ostatnými jednoduchými tagmi, ktoré bude snímať na bráne pripevnená UHF čítačka s optimálne nastaveným dosahom.

2.3.2 Obecná schéma

Protokol sa skladá z dvoch hlavných častí a to z registrácie, viď. obrázok 2.8 a samotnej autentizácie, viď. obrázok 2.9.



Obr. 2.8: Registračná časť autentizačného protokolu

Registrácia SAMu

Účelom registrácie je zaradiť SAM do systému tak, aby mohol komunikovať s registrovanými čipovými kartami. V procese sú na SAM nahrávané tajné kľúče potrebné k funkcii systému a identifikátory. Postup je nasledovný.

1. Registračný server si vygeneruje náhodný kľúč $K_{CS} \in_R \{0,1\}^{256}$, ak ho už nevygeneroval v minulosti.

2. Vygeneruje súkromný kľúč a pre SAM, pričom $a \in_R Z_q$. Z neho odvodí verejný kľúč A tak, že vynásobí generujúci bod G eliptickej krivky so súkromným kľúčom SAMu a, teda $A = aG$. Tieto parametre sú využité pri dohode spoločného symetrického šifrovacie kľúča.
3. Náhodne určí ID_R SAMu, pričom $ID_R \in_R \{0,1\}^{64}$ a vypočíta jeho symetrický kľúč $K_{R1} = H(K_{CS}, ID_R)$, ktorý slúži k účelu autentizácie čipovej karty.
4. Nakoniec vygenerované informácie odošle. Pre SAM posiela ID_R, K_{R1}, a, A.

Registrácia čipovej karty

Registrácia čipovej karty prebieha podobne ako registrácia SAMu, pričom sú na kartu nahrané potrebné informácie a karta je spolu k nej priradenými UHF tagmi uložená do rozšíriteľnej databáze systému. Registrácia sa skladá z nasledujúcich krokov.

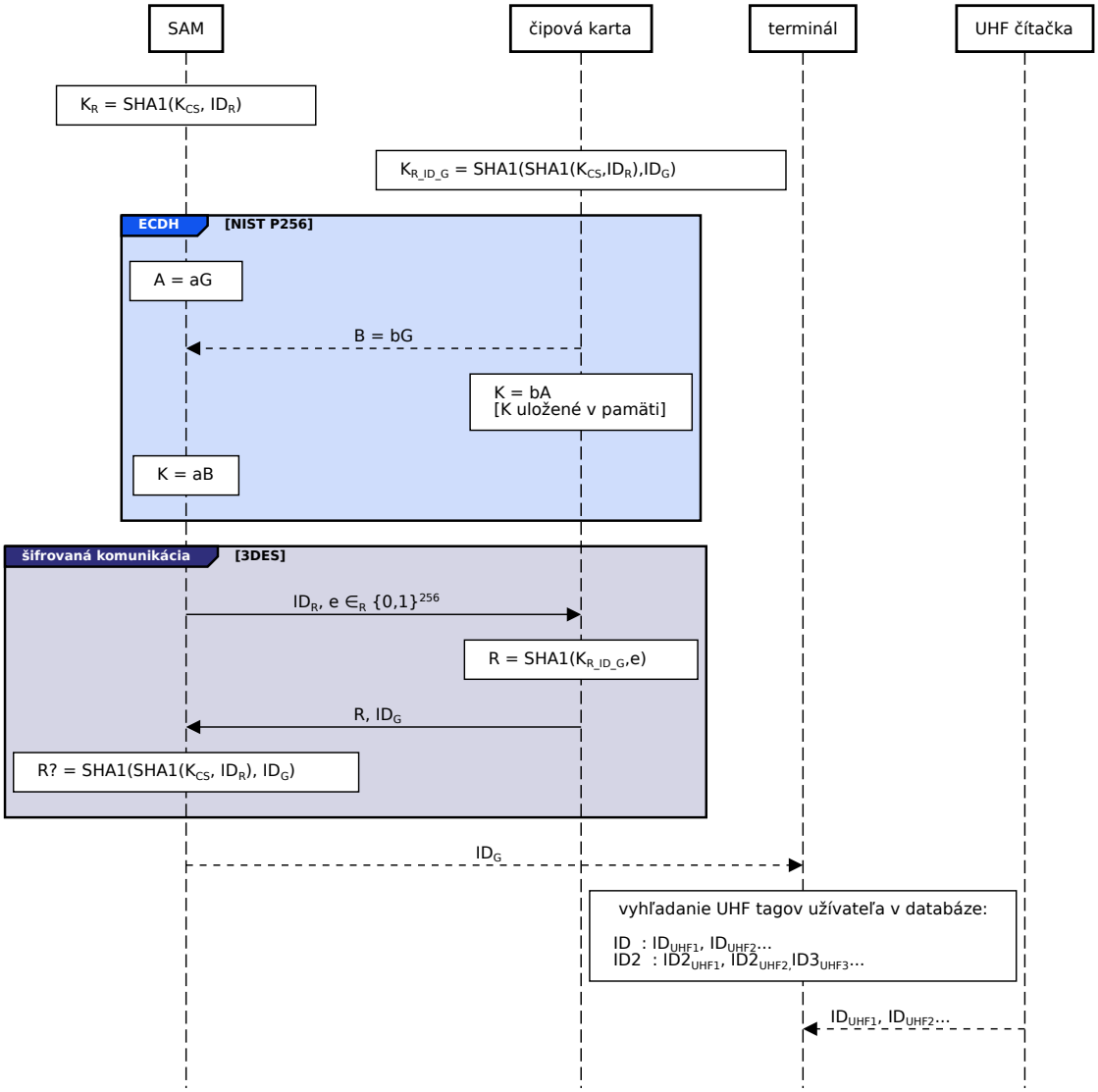
1. Registračný server priradí ID čipovej karte, pričom $ID_{G1} \in_R \{0,1\}^{64}$. Identifikátory sa priraďujú inkrementálne.
2. Spočíta hash $K_{R\_ID\_G1} = H(K_{R1}, ID_{G1})$.
3. Čipovej karte zašle vygenerované informácie $ID_{G1}, K_{R1}, K_{R\_ID\_G1}, A$.
4. Registračný server tiež pre čipovú kartu predgeneruje určitý počet šifrovacích kľúčov, a to tak, že vygeneruje náhodné $b1, b2, b3.. \in_R Z_q$, vypočíta verejný kľúč na eliptickej krivke $B1 = b1G, B2 = b2G, B3 = b3G$. Za použitia verejného kľúča SAMu, predpočítaného v jeho registrácii a privátnych kľúčov čipovej karty vypočíta šifrovacie kľúče, $K1 = b1A, K2 = b2A, K3 = b3A...$
5. Šifrovacie kľúče a verejné kľúče server odošle čipovej karte.
6. Užívateľ si zvolí počet UHF tagov, ktoré budú patriť k danej čipovej karte a tieto pomocou UHF čítačky naskenuje a uloží do databáze uloženej v terminály.

Autentizácia

Hlavnými bodmi autentizácie je Diffie-Hellman založený na Eliptickej krivke ECDH (Elliptic Curve Diffie-Hellman), ktorý slúži na ustanovenie spoločného symetrického kľuča a samotná šifrovaná komunikácia, ktorá zaistí autentizáciu užívateľa.

Ustanovenie symerického kľúča - ECDH

1. Čipová karta náhodne vyberie sadu verejného a šifrovacieho kľúča, uložených pri registrácii. Verejný kľuč zašle SAMu a šifrovacím kľúčom je nasledovne šifrovaný celý zvyšok komunikácie.
2. SAM prijme verejný kľúč B, patriaci čipovej karty a vypočíta symetrický šifrovací kľúč $K_a = K_b = aB$.



Obr. 2.9: Autentizačná časť

3. Zvyšok komunikácie je šifrovaný symetrickou šifrou za využitia vypočítaných kľúčov $K_a = K_b$.

Samotné overenie užívateľa

1. SAM vygeneruje náhodné číslo $e \in_R \{0,1\}^{256}$. Toto spolu s jeho identifikátorom ID_R zašle čipovej karte.
2. Čipová karta vyparsuje prijaté e, vypočíta odpoveď $R = H(K_{R\_ID\_G1}, e)$ a odošle ju späť, spolu so svojim ID_{G1}.
3. SAM rekonštruuje odpoveď R, či $R \stackrel{?}{=} H(H(K_R, ID_{G1}), e))$ a porovná jú s odpoveďou od čipovej karty.
4. Ak sa R vypočítané na strane čipovej karty rovná R vypočítanému na strane SAMu, SAM predá terminálu ID čipovej karty.

5. Terminál priradí k prijatému ID_{G1} identifikačné čísla UHF tagov a dá výzvu UHF čítačke na začatie skenovania.
6. Po naskenovaní všetkých tagov v dosahu UHF čítačky sú tieto porovnané s tagmi získanými z databáze. Ak je užívateľ úspešne autentizovaný a zároveň disponuje všetkými nutnými UHF tagmi je mu prístup udelený. V opačnom prípade je mu prístup zamietnutý.

2.3.3 Výber hardvéru a obmedzenia s ním spojené

S ideálnou podporou kryptografických operácii na slabom hardvéri aký obsahujú čipové karty by protokol využíval nasledovné parametre, podľa doporučení NIST.[30]

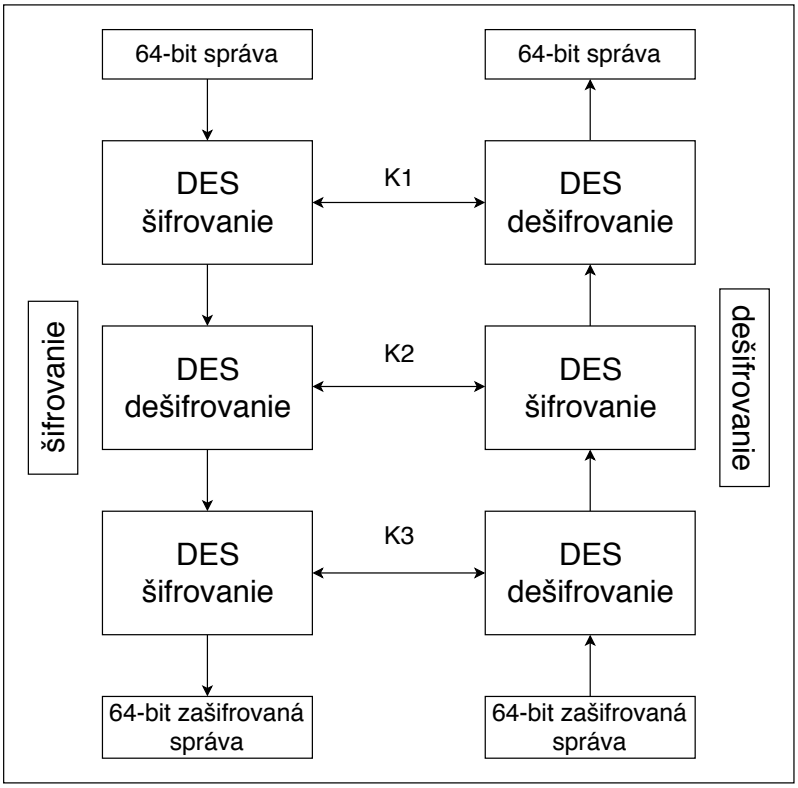
- Eliptická krivka - NIST P256,
- bloková šifra - AES 256,
- hešovacia funkcia - SHA 256.

Nie všetky implementácie čipových kariet obsahujú potrebné kryptografické primitíva a komunikačné rozhrania a preto musel protokol ustúpiť obmedzeniam. Problémom je najmä implementácia primitív nad eliptickými krivkami, či silnejšie symetrické šifry.

SAM

Hlavnou úlohou SAMu je komunikácia s čipovou kartou užívateľa a bezpečné uloženie kľúčov. Najmä kvôli uloženiu kľúčov bola ako SAM vybraná kontaktná čipová karta s operačným systémom MULTOS, ktorá je považovaná za tamper-proof, čiže nesfalšovateľné zariadenie. Konkrétne ide o MULTOS ML4 s verziou MULTOSu 4.2.1. Čipová karta má hardvérovo implementované operácie na eliptickej krivke, konkrétne zvolená verzia podporuje aj krivku o veľkosti 256b, ktorá je uvedená v návrhu kryptografických primitív. Pre ECDH je nutné násobiť body na eliptickej krivke, čo je tiež v konkrétnej čipovej karte implementované.

Konkrétny MULTOS nemá podporu AESu, teda bolo potrebné zvoliť inú využiteľnú blokovú šifru. Ďalšia perspektívna bloková šifra je 3DES, ktorý berie 24 bajtový kľúč a šifruje bloky dát po 8 bajtoch. 3DES síce v konkrétnom prevedí MULTOSu taktiež nie je podporovaný, dokáže byť však vyskladaný s obyčajného DESu, ktorý podporu v hardvéri má. Pre šifrovanie bol teda použitý DES v poradí šifrovanie-dešifrovanie-šifrovanie, vždy za použitia iných 8 bajtov z celkového 24 bajtového kľúča, čím bol imitovaný protokol 3DES. Na dešifrovanie bol využitý DES v poradí dešifrovanie-šifrovanie-dešifrovanie takisto za použitia vždy 8 bajtov z celkových 24 šifrovacieho kľúča, v tomto prípade bol však kľúč použitý v opačnom poradí, viď. obrázok 2.10.



Obr. 2.10: Blokové schéma 3DES

Podporu hešovacích funkcií má daný MULTOS pomerne obmedzenú, preto jediná možná varianta bola použiť funkciu SHA-1, ktorej výstupom je 20 bajtový heš.

Softvér čipovej karty je možné programovať v Jave, C či špeciálnom MULTOS assemblery. Pre túto prácu bol zvolený programovaný jazyk C v kombinácii s MULTOS assemblerom.[31]

Užívateľská čipová karta

Čipová karta musí byť jednoduchá a užívateľsky priateľská a preto bola určená ako jeden z jej potrebných atribútov bezkontaktnosť. Vybraná bola JavaCard NXP JCOP J3A081, s verziou operačného systému 2.2.2. Táto karta má duálne rozhranie, čo znamená, že dokáže komunikovať ako aj kontaktne, tak aj bezkontaktne, a zároveň vykazuje najlepšie výpočetné schopnosti zo súčasných čipových kariet. Karty JavaCard sú štandardne programovateľné programovacím jazykom Java, ktorý je dodatočne prispôsobený na nízky výkon procesoru karty. Keďže pôvodný návrh autentizačného protokolu proponoval počítať šifrovací kľúč - teda ECDH priamo na čipovej karte, pri výbere vhodného hardvéru bol braný do úvahy i tento parameter.

Na vybranej karte bol spustený test dostupných kryptografických primitív JCAlgTest, ktorý je dostupný na GitHube. [33] Výsledky ukázali, že karta síce podporuje

operácie na 256b eliptickej krivke, nie však všetky potrebné na vykonanie ECDH, viď. výpis2.5.

Výpis 2.5: Časť výpisu z JCAlgTest

```
javacard.security.KeyPair ALG_EC_FP on-card generation
ALG_EC_FP LENGTH_EC_FP_192;yes;0.804000
ALG_EC_FP LENGTH_EC_FP_224;error(ILLEGAL_VALUE);
ALG_EC_FP LENGTH_EC_FP_256;error(ILLEGAL_VALUE);
javacard.security.KeyBuilder
TYPE_EC_FP_PRIVATE LENGTH_EC_FP_192;yes;0.218000
TYPE_EC_FP_PRIVATE LENGTH_EC_FP_224;yes;0.218000
TYPE_EC_FP_PRIVATE LENGTH_EC_FP_256;yes;0.218000
```

Z testu vyplýva, že karta nieje schopná generovať verejný a súkromný kľúč potrebný pre ECDH na 256b eliptickej krivke. Vznikli dva varianty, ako tento problém obísť a to používať eliptickú krivku nad 192b konečným poľom, alebo generovať kľúče potrebné k šifrovanej komunikácii mimo čipovú kartu. Použitie nižšej krivky by znížilo bezpečnosť protokolu a navyše predom generované kľúče pomerne zmenšujú čas potrebný na autentizáciu, čo je pri efektívnosti protokolu značná výhoda, preto bol vybraný variant generovania kľúčov predom a ich nahranie na kartu v čase jej registrácie.

Terminál (autentizačný server)

Terminál prenáša komunikáciu medzi JavaCard a MULTOSom a zároveň ukladá databázu registrovaných čipových kariet a ku nim pripojených UHF tagov. Doležitým faktorom pre terminál je nízka spotreba a malé rozmery. Terminál by mal zároveň mať podporu zasielania a prijímania APDU správ, ktoré sú nosným prvkom informácií prenesených medzi čipovými kartami.

Vybraný bol nenáročný počítač RaspberryPi B2 s 1GB RAM pamäte, ktorý má 32b procesor postavený na ARM architektúre a beží na ňom operačný systém Raspbian.

Registračný server

Pre základnú demonštráciu autentizačného systému bolo ako registračný server vybraté zariadenie RaspberryPi, kvôli zlúčenej prevádzke spolu s terminálom. Server je však jednoducho prenositeľný na iné zariadenie s operačným systémom založeným na Linux Debian, a teda je možné ho prevádzkovať na výkonnejšom a samostatnom zariadení.

Zhrnutie vybraných kryptografických primitív

Po zistení dostupnosti rôznych operácií na čipových kartách boli vybrané nasledovné kryptografické primitíva, ktoré budú používané v autentizačnom protokole. Výber bol jasne daný hardvérovými možnosťami daných kárt.

- Eliptická krivka - NIST P256,
- bloková šifra - 3DES s 192 bitovým kľúčom,
- hešovacia funkcia - SHA-1.

2.3.4 Implementácia autentizačného protokolu

Výber programovacích jazykov pre autentizačný systém je zhrnutý v tab. 2.4.

Tab. 2.4:

časť systému	programovací jazyk	vývojové prostredie
JavaCard	Java	NetBeans
MULTOS	C, MULTOS assembler	Eclipse
UHF	Python	VIM
registračný server	Python	VIM
autentizačný server	Python	VIM

Vývojové prostredia pre čipové karty a všetky náležitosti potrebné k správe aplikácií čipových kariet boli nainštalované a spúšťané vo virtualizovanom operačnom systéme Windows 7.

Programovanie na JavaCard

Vývoj aplikácie prebiehal v upravenom prostredí NetBeans. Build prebiehal pomocou programu `ant`, ktorému bol nutne špecifikovaný build súbor `build.xml`, ktorý obsahoval umiestnenie SDK a umiestnenie výstupu buildu. Takto bol vygenerovaný súbor `*.cap`, ktorý bol priamo nahrateľný na čipovú kartu. Pre nahranie aplikácie na kartu bola využitá aplikácia GPShell.exe, ktorej bolo špecifikované napríklad ID aplikácie a kľúč pre nahrávanie na kartu.

V implementácii samotného protokolu na čipovú kartu JavaCard boli využívané najmä triedy pre šifrovanie a hešovanie obsiahnuté v SDK, konkrétne `javacard.security.DESKey` pre uloženie 3DES kľúča, `javacardx.crypto.Cipher` pre uskutočnenie samotného šifrovania a dešifrovania pomocou 3DES a `javacard.security.MessageDigest`, odkiaľ bola využitá funkcia pre hešovanie SHA-1. Príklad metódy

na hešovanie funkciou SHA1 je uvedený na výpise 2.6. Do metódy vstupujú dva argumenty typu `byte[]`, z ktorých spojenia vznikne výsledný heš a argument `byte[] res`, do ktorého je uložený výsledok hešovania. Samotné hešovanie na JavaCard priebeha tak, že sa založí instancia triedy `MessageDigest`, v ktorej je určený typ hešu ako SHA1. Do tejto instancie sa uloží prvé pole bajtov a zavolaním metódy `doFinal()` sa do nej uloží druhé pole bajtov, špecifikuje sa výstupná premenná a do nej sa uloží výsledok.

Výpis 2.6: Metóda na hešovanie pomocou SHA1, JavaCard.

```
public void shaHash(byte[] first,byte[] second,byte[] res)
{
    MessageDigest hash = MessageDigest.getInstance
    (MessageDigest.ALG_SHA,false);
    hash.update(first,(short) 0,(short) first.length);
    hash.doFinal(second,(short) 0,(short) second.length,
    res, (short) 0);
}
```

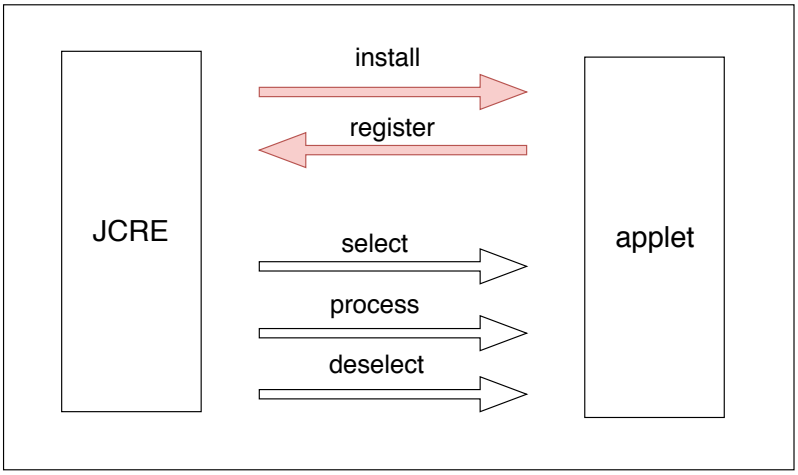
Stavba JavaCard appletu. Hlavná trieda musí importovať balík `javacard.framework`, ktorý obsahuje triedy a rozhrania nevyhnutné pre vývoj JavaCard appletu. Definuje aj triedu `javacard.framework.Applet`, ktorá určuje komunikáciu s JCRE a napríklad aj jednobajtové konštanty hodnôt v hlavičke APDU príkazu. Hlavná trieda každého appletu musí dediť práve z triedy `javacard.framework.Applet`.

Applet musí obsahovať metódu `install`, ktorá je hlavným vstupom appletu. V nej je klasicky vytvorená instancia hlavnej triedy. V jej konštruktore je potom volaná metóda `register`, ktorá daný applet registruje u JCRE.

Po inicializovaní applet čaká kým bude vybraný APDU príkazom `select`. Vtedy je volaná metóda `process`, ktorá obsluhuje celý zvyšok komunikácie s daným appletom. Parametrom tejto metódy je `APDU`, teda instancia triedy `javacard.framework.APDU` spracovávajúca príchodzie a odchodzie APDU. Proces komunikácie medzi JCRE a appletom je znázornený na obrázku 2.11. [37]

V metóde `process` je porovnávaný APDU bajt CLA s typom inštrukčnej sady obsiahnutej na danej karte. Neskôr je vyhodnocovaný bajt INS v inštrukcii `switch`. Takto sa určí ktorá časť programu bude vykonaná po inštrukcii INS.

Pre odoslanie spracovaných bajtov naspäť k čítačke je nutné vytvorenú instanciu triedy `javacard.framework.APDU` označiť ako odchodziu metódou `setOutgoing()`, určiť počet odchodzích bajtov metódou `setOutgoingLength()` a nakoniec ich odoslať pomocou `sendBytes()` alebo `sendBytesLong()`.



Obr. 2.11: Proces komunikácie medzi JCRE a appletom, JavaCard

Vrámci autentizačného systému je časť na JavaCard obsiahnutá v jednej triede, ktorá zahŕňa všetky potrebné metódy na šifrovanie, dešifrovanie, generovanie náhodného čísla, či hešovanie a zároveň hlavnú `switch` podmienku, ktorá spravuje príchodzie a odchodzie APDU správy.

Keďže pri registrácii JavaCard ako čipovej karty sa na ňu nahrávajú šifrovacie kľúče dopredu, nie je nutné na nej vykonávať akékoľvek operácie na eliptickej krivke. Kľúče uložené v poli sú potom pri jednotlivých autentizáciach náhodne vyberané. Kvôli obmedzenému množstvu dostupných šifrovacích kľúčov je vhodné čipovú kartu po určitej dobe preregistrovať, a to z toho dôvodu, že pri viacnásobnom použití jedného kľúča nie je zaručená nevystopovateľnosť užívateľa pri odchytení komunikácie pri autentizácii.

Programovanie na Multos

SmartDeck je prostredie pre vývoj aplikácií určených pre čipové karty MULTOS. Je dodávaný priamo spoločnosťou MULTOS. Ide o grafické užívateľské prostredie na vývoj v jazyku C a assembleri zahrnuté do Eclipse IDE. SmartDeck je možné po registrácii stiahnuť na oficiálnych stránkach spoločnosti.

SmartDeck poskytuje vstavaný kros-kompilátor a takisto simulátor prostredia čipovej karty bežiaci na počítači. Po kompilácii programu priamo v SmartDeck vznikne sputiteľný súbor `*.hzx`. Tento súbor je nutné programom `halugen` obsiahnutým v SDK konvertovať na výstupný súbor s príponou `*.alu`(application load unit), ktorý je možné nahrať na kartu. SDK je rovnako ako SmartDeck stiahnuteľné na stránkach výrobcu.

Pre nahranie aplikácií na kartu je možné použiť program MUtil s grafickým užívateľským rozhraním. Pre nahranie je nutné v ňom špecifikovať skrze ktorú čítačku

z dostupných má MUtil komunikovať. V záložke Load Test je nutné vybrať `*.alu` súbor, ktorý bude na kartu nahraný. Ďalej je potrebné vyplniť ID appletu. Nahratie prebehne stlačením tlačítka Load alebo Reload. Rozšírené možnosti v záložke Load Unit umožňujú upraviť tzv. `access list`, kde je bližsie nastaviteľné napríklad aké komunikačné rozhranie karta používa, alebo či má zapnuté silné kryptografické funkcie. Tak ako aj vyššie spomenuté pomôcky aj MUtil je stiahnuteľný na stránkach spoločnosti MULTOS.[34]

MULTOS poskytuje presnú dokumentáciu, pričom jedným z najdôležitejších dokumentov pri písaní aplikácie je MULTOS Developer's Reference Manual. Tu sú zhrnuté všetky funkcie, ktoré karty MULTOS vo všeobecnosti podporujú, konkrétnu podporu danej karty je potom nutné vyhladať na stránkach daného výrobcu. S pomocou MDRM potom vznikajú funkcie ako napríklad pre hešovanie SHA-1, viď. výpis 2.7.

Ďalším dôležitým dokumentom je MULTOS Developer's Guide, ktorý popisuje schému a funkčnosť MULTOS čipových kariet.

Spoločnosť MULTOS tiež poskytuje SDK, ktoré obsahuje knižnice zjednodušujúce prácu s assemblerom, či definujúce funkcie pre jazyk C. Zahŕňa tiež konštanty potrebné pre prácu s príchodzím APDU príkazom.

Výpis 2.7: Operácia SHA-1 na čipovej karte MULTOS

```
__push((BYTE*)len); //na zasobnik sa prida ukazovatel
                    //na dlzku vstupu
__push(output);     //pole vstupnych dat
__push(input);      //pole vystupnych dat (SHA-1 -> 20B)
__code(PRIM, 0xca); //instrukcia na spracovanie
                    //na zasobniku ulozenych dat
```

Stavba MULTOS appletu. Pre funkčnosť appletu je nutné definovať jeho ID a takisto nastaviť záznam o applete v Directory File, viď. výpis 2.8.

Výpis 2.8: Nastavenie aid a dir

```
#pragma attribute("aid", "f0␣00␣00␣01")
#pragma attribute("dir",
 "61␣10␣4f␣4␣f0␣00␣00␣01␣50␣8␣65␣6c␣6f␣79␣61␣6c␣74␣79")
```

Ďalej je nutné obsiahnuť deklarácie hlavičkových súborov potrebných pre písanie programu, ako napríklad `#include <multoscomms.h>`. Pre lepšiu čiteľnosť programu je štandardné zadefinovať konštanty k hexadecimálne uvedeným inštrukciám, či chybovým hláškam.

Dôležitá je práca s pamäťou. Informácie je možné ukladať do volatilnej RAM, či statickej EEPROM obsiahnuté v karte. Do RAM pamäte sa ukladajú odchodzie a príchodzie APDU správy, pričom iniciácia premenných patriacich sem musí byť vymedzená pragmou `#pragma melpublic`. Do pamäti RAM je možné ukladať i dáta, ktoré sú určené k zotrvaniu na karte a nieje žiaduce ich posielať z karty ako APDU správy. Tieto dáta sú deklarované v časti `#pragma melsession`. Dáta potrebné uchovať aj po odpojení karty od napájania je nutné iniciovať v priestore vymedzenom ako `#pragma melstatic`.

Po prijatí APDU kartou nastáva kontrolovanie bajtu CLA, pričom inštrukčná sada v ňom uvedená musí súhlasiť s inštrukčnou sadou na karte. V ďalšom kroku sa spracúva bajt INS vo výraze `switch`, ktorá rozradí prichádzajúce dáta k tej časti programu, ktorá ich má interferovať.

Pre odoslanie spracovaných dát je využívaná knižničná funkcia `ExitSW()`, so špecifikovaným chybovým hlásením v prípade nesprávneho spracovania inštrukcie, alebo funkcia `ExitLa()` pre odoslanie dát. V nej sa špecifikuje počet bajtov určených na odoslanie.

Prvým nutným krokom časti autentizačného protokolu obsiahnutého na MULTOS čipovej karte je vypočítanie šifrovacieho symetrického kľúča pomocou ECDH. V statickej pamäti karty sú definované doménové parametre eliptickej krivky NIST P256. Ide o pole shortov, ktoré obsahuje nasledujúce parametre:

1. formát doménových parametrov, konkrétne hodnotu 0x00,
2. veľkosť eliptickej krivky v bajtoch,
3. prvočíslo, teda veľkosť konečného poľa F_p nad ktorým je krivka postavená,
4. parameter $a \in F_p$,
5. parameter $b \in F_p$,
6. súradnice X a Y,
7. rád generujúceho bodu krivky,
8. kofaktor.

Po špecifikovaní doménových parametrov je možné krivku používať v kryptografických inštrukciách čipovej karty, teda napríklad pri násobení bodu na krivke, ktoré sa používa pre dohodnutie tajného šifrovacie kľúča ako je uvedené na výpise 2.9.

Výsledok operácie `PRIM_EC_MUL`, ktorá vynásobí bod na krivke s číslom je takisto bod na krivke, teda pri 256b krivke ide o 512b (256b pre súradnicu X, 256b pre súradnicu Y), čiže 64B číslo. Pre MULTOS je nutné špecifikovať formu zadania bodu na krivke, čo pridáva k veľkosti bodu ďalší bajt, ktorý však neovplyňuje jeho samotnú hodnotu. Pre zadanie bodu vo forme súradnica X, súradnica Y má bajt špecifikújúci formu hodnotu 0x04. Pre šifrovanie pomocou 3DES je nutný kľúč o dĺžke 24B, teda je použitých prvých 24B z výsledku násobenia bodov spomenutého vyššie.

Výpis 2.9: Násobenie bodu na eliptickej krivke, MULTOS

```
//na zasobnik sa postupne pridavaju polia bajtov:
__push(ecDomainParams256);  //domenove parametre EC 256b
__push(public_point_B);     //verejený kluc JavaCard
__push(private_prime_a);    //sukromny kluc MULTOS
                            //33 bajtove cislo
__push(DH_transaction_key); //pre ulozenie vysledku
__code(PRIM, PRIM_EC_MUL);  //volanie samotnej operacie
                            //PRIM_EC_MUL = 0xD4
```

Pre šifrovanie bola vytvorená funkcia, ktorá volá inštrukciu na vykonanie DESu 3x po sebe a to vždy s inou časťou kľúča tak, aby bola výsledná šifra kompatibilná so štandardizovaným 3DESom používaným na strane JavaCard.

UHF časť. UHF čítačka UHF Reader 18 je ovládaná pomocou externej, prispôsobenej `Python` knižnice UHFReader18. Celá časť systému týkajúca sa UHF je teda takisto napísaná v jazyku `Python`. Po vytvorení objektu UHFReader18() a spojení s čítačkou na comporte, ku ktorému je čítačka pripojená (štandardne /dev/ttyUSB0) je možné čítačku zapnúť, vypnúť, nastaviť silu vysielania, zapnúť či vypnúť bzučiak signalizujúci načítanie tagu, viď. výpis 2.10.

Výpis 2.10: Nastavenie UHF čítačky

```
uhfr=UHFReader18()                          #vytvorenie objektu
uhfr.openPort("/dev/ttyUSB0", 57600)        #pripojenie citacky

uhfr.setWorkMode(uhfr.MODE_ACTIVE)          #nastavovaci mod
uhfr.setProto(uhfr.PROTO_6C)                #ISO18000-6C
uhfr.setInterface(uhfr.IFACE_SERIAL)        #RS232 komunikacia
uhfr.setStorageArea(uhfr.STORAGE_EPC)       #tagy budu EPC tagy
uhfr.setPower(10)                           #mozny rozsah 0-30
uhfr.setBuzzer(uhfr.ON)                     #zapnutie bzuciaku
uhfr.setWorkMode(uhfr.MODE_ANSWER)          #skenovaci mod
```

Následne je využívaná funkcia `tagQuery()` na skenovanie tagov, ktorá vráti UID tagu. Funkcia je využívaná na registráciu tagov k čipovej karte, či vyhľadávanie naskenovaných tagov v databáze a ich porovnávanie. Nevýhodou konkrétnej čítačky je fakt, že tagy dokáže korektne načítavať až po úplnom nastavení, a teda ak je tag naskenovaný v procese nastavovania, čítačka jeho ID vyhodnotí ako nečakaný sled bajtov a dostane sa do chybového stavu.

Zostavovanie systému

Pre program pracujúci mimo čipových kariet, teda v terminále a registračnom serveri bol zvolený programovací jazyk Python3. Hlavným atribútom pre výber programovacieho jazyka bola jednoduchá manipulácia s APDU správami smerujúcimi od a ku čipovým kartám. Python pre túto potrebu obsahuje modul `pyscard`.[35]

`Pyscard` ponúka podporu pre vytváranie spojení medzi počítačom a čítačkou čipových kariet a taktiež poskytuje prehľadné API pre posielanie a prijímanie APDU správ.

Ďalším dôležitým predpokladom pre výber programovacieho jazyka bola podpora kryptografických primitív a hešovacích funkcií. Pre hešovanie `Python` ponúka napríklad modul `hashlib`, ktorý poskytuje veľa variácií hešovacích funkcií a teda aj pre autentizačný protokol potrebnú SHA-1.

Pre operácie na eliptickej krivke je dostupný modul `fastecdsa`, ktorý má podporu, mimo iných, aj krivky NIST P256, takisto potrebnej na autentizačný protokol.

Registrácia Registráciu zabezpečuje program `registration.py`. Pomocné funkcie sú obsiahnuté vo vytvorenom module `SmartCardsModul.py`.

Pri prvom spustení programu registračného skriptu je nutné vygenerovať symetrický kľúč pre registračný server. Po vygenerovaní je kľúč uložený do súboru `K_CS` v priečinku `/srv/databases/`. Pri generácii registračného kľúča je vždy kontrolované, či súbor `K_CS` už existuje, a ak áno kľúč je načítaný priamo z neho a teda nie je znovu generovaný.

Ďalej program vyzve užívateľa otázkou, či už existuje registrovaný SAM, ku ktorému je potrebné doregistrovať kartu. Ak nie je, a teda bude nahraný ako aj SAM, tak i čipová karta, vygenerujú sa parametre potrebné na fukciu SAMu, viď. výpis 2.11.

Výpis 2.11: Výpis z interaktívneho skriptu registrácie - generovanie dát SAMu

```
SAM ALREADY GENERATED  yes/no                                          1
>>>no                                                                  2
ENTER NUMBER OF UHF TAGS CONNECTED TO SMARTCARD                        3
>>>2                                                                   4
SERVER KEY LOADING                                                     5
SAM CRED GENERATION                                                    6
ID OF SAM IS: '[13, 98, 24, 198, 253, 213, 173, 9]'                    7
```

V zložke `/srv/databases/` je vytvorený súbor, ktorého názov obsahuje ID práve registrovaného SAMu, teda napríklad `'[13, 98, 24, 198, 253, 213, 173, 9]'`. V takomto súbore je uložený jeho verejný kľúč, ktorý je potrebný pre následnú generáciu spoločných kľúčov pre užívateľskú čipovú kartu. Keďže verejný kľúč je v

tomto prípade bod na eliptickej krivke, bolo nutné využiť funkciu `Python` modulu `fastecdsa`, ktorá dokáže exportovať a importovať body vo formáte ASN.1 (Abstract Syntax Notation One), viď. výpis 2.12.

Výpis 2.12: Verejný ECDH kľúč v ASN.1 formáte

```
-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAELj3UxmaooeC7WBOUBqTA
cRzoN3hW2n1dvD5OKavByq0y1lmG5H6TAhPijVSeIimAHLMMX2RkQg9H
FVztb+B1A==
-----END PUBLIC KEY-----
```

Vygenerované parametre sú následne odoslané SAMu, teda MULTOSu, ktorý si hodnoty vykopíruje do svojich premenných.

Ak už zaregistrovaný SAM existuje, a je potrebné vygenerovať parametre iba pre čipovú kartu, užívateľ je vyzvaný zadať ID SAMu, podľa ktorého je dohľadaný jeho verejný kľúč, viď. výpis 2.13.

Výpis 2.13: Výpis z interaktívneho skriptu registrácie - zadanie ID SAMu

```
SAM ALREADY GENERATED  yes/no
>>>yes
INSERT SAM ID
>>>[13, 98, 24, 198, 253, 213, 173, 9]
ENTER NUMBER OF UHF TAGS CONNECTED TO SMARTCARD
>>>2
SERVER KEY LOADING
SAM CRED LOADING
ID OF SAM IS: '[13, 98, 24, 198, 253, 213, 173, 9]'
```

Pomocou vyhľadaného kľúča sú následne vygenerované parametre pre čipovú kartu, teda jej privátne a verejné kľúče pre ECDH, tajný kľúč získaný ECDH, jej ID a zahešované ID SAMu, ku ktorému patrí. ID čipovej karty je priraďované inkrementálne, posledné priradené ID je vždy uložené v súbore `/srv/databases/last_registered`. Tieto parametre sú následne odoslané na čipovú kartu. ECDH kľúče sú z dôvodu ich početnosti posielané zvlášť, teda aj Javacard ako čipová karta ich pri prijatí spracováva inak ako zvyšok potrebných parametrov.

Keďže do jednej APDU správy je možné uložiť 255 bajtov dát a jedna sada posielaných kľúčov má vždy 96 bajtov (64 bajtov pre verejný kľúč a 32 bajtov pre digitálny kľúč využívaný na šifrovanie komunikácie), je možné v jednej správe odoslať najviac dve celistvé sady. Pre poslanie 200 sád kľúčov sa teda v stokrát opakovanom cykle vyšle APDU príkaz s rovnakým číslom inštrukcie, avšak s inými dátami na

JavaCard. JavaCard si drží čítač prijatých sád kľúčov a tieto postupne ukladá do jednorozmerného poľa.

Čas potrebný na registráciu je priemerne **26.68 s**, zahŕňajúc nahranie parametrov na MULTOS a nahranie parametrov a 200 sád kľúčov na JavaCard, nie však načítanie UHF tagov, ktoré sa môže líšiť v závislosti na ľudskej chybe skenovania.

Načítanie tagov prebieha tak, že na začiatku registrácie užívateľ zadá počet UHF tagov, ktoré budú k danej čipovej karte patriť a po registrovaní čipovej karty ich po jednom naskenuje, viď. výpis 2.14.

Výpis 2.14: Výpis z interaktívneho skriptu registrácie - UHF časť

```
UHF READER SET SUCESSFULLY
PLEASE SCAN 2 UHF TAGS NOW
'e28011700000002076a68c8ac' scanned tag
1 tags to scan
'e28011700000002076a68c8ae' scanned tag
0 tags to scan
UHF reader powered off
```

Dvojica UHF tag - ID čipovej karty sa vždy uloží do SQL databáze, ktorú obsluhuje vytvorený modul `SQLModul.py`, založený na štandardizovanom databázovom ovládači `connector`.

Databáza môže bežať na stroji umiestnenom v lokálnej sieti spolu s autentizačným a registračným serverom tak, aby oba boli schopné s databázou pracovať. Takisto je možné SQL databázu prevádzkovať priamo na autentizačnom serveri, teda termináli (v konkrétnom prípade na zariadení RaspberryPi), vďaka programu `mysql-server-default`, dostupnom pre zariadenia s operačným systémom založeným na Debian Linux. Pre demonštratívne riešenie bol zvolený variant s databázou činnou na autentizačnom serveri.

Autentizácia Keďže autentizačný protokol beží na zariadení RaspberryPi s operačným systémom Rasbpian, ktorého init systémom je tak ako pri mnohých iných linuxových distribúciach `systemd`, mohla byť pre spúšťanie autentizačného programu vytvorená `systemd` servisa.

Súbory obsahujúce autentizačný program sú umiestnené v zložke `/usr/local/bin/`, ktorá je obsiahnutá v premennej `PATH` a teda program bude spustiteľný z akéhokoľvek východzieho adresára.

Pre servisu bol vytvorený servisný súbor v `/etc/systemd/system/` s názvom `authentication.service`. Tu je špecifikovaná funkcia servisy, prípadne jej systémové chovanie. Systemd má za úlohu spúšťať servisu pri štarte zariadenia a sprístupniť jej jednoduché manuálne ovládanie. Pri prípadnom excese servisy ju opäť

spustí. Pravidlá v servisnom súbore sú zobrazené vo výpise 2.15. Servisa je potom jednoducho spustiteľná príkazmi `systemctl enable authentication` a `systemctl start authentication`. Výpisy zo servisy je možné sledovať použitím príkazu `sudo journalctl -f -u authentication`. Časť výpisov servisy je zobrazená na výpise 2.16.

Výpis 2.15: Servisný súbor servisy authentication.service

```
[Unit]
Description=Authentication protocol
After=multi-user.target

[Service]
Type=simple
ExecStart=/usr/local/bin/authentication.py
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

Samotný proces sprostriedkovávania autentizácie medzi MULTOSom a JavaCard je opakovane volaný v cykle, v hlavnom autentizačnom `Python` programe nazvavom `authentication.py`. Tak ako registračný skript i tento používa pomocné funkcie zabezpečujúce jednoduchšiu tvorbu APDU správ, či prácu s bajtmi, ktoré sú obsiahnuté vo vytvorenom module `SmartCardsModul.py`.

Výpis 2.16: Čiastkový výpis výstupu servisy authentication.service

```
pi python3[6176]: apps selected: jc: '0x90',multos: '0x90'
pi python3[6176]: sending first challenge to JC
pi python3[6176]: sending pub key B (JC -> to M)
pi python3[6176]: 0x28 bytes to receive,0x90, 0x0
pi python3[6176]: sending ID_R1 and random e (M -> to JC)
pi python3[6176]: sending R, computed in JC (JC -> to M)
pi python3[6176]: 0x8 bytes to receive, 0x90, 0x0
pi python3[6176]: sending ID_G1 (M -> to terminal (pi))
pi python3[6176]: smartcards disconnected
pi python3[6176]: ID of JC: 14
pi python3[6176]: ---smartcard authentication success---
pi python3[6176]: tags to scan:
['e2801170000002076a68c8ac','e2801170000002076a68c8ae']
```

Po priložení JavaCard na čítačku je jej poslaný APDU príkaz pre zahájenie autentizácie. Na tento príkaz odpovedá vybraním šifrovacieho kľúča a zaslaním k nemu patriaceho verejného kľúča späť k `Python` programu. Ten prijatú APDU odpoveď upraví a vytvorí z nej APDU príkaz smerovaný k MULTOSu. Úprava prebehne odstránením dvoch posledných bajtov SW, ktoré hovoria o korektnom alebo chybovom spracovaní príkazu, a pridaní APDU hlavičky k zostávajúcim dátam.

Takýmto spôsobom `Python` program preposiela komunkáciu, až do finálnej odpovedi MULTOSu. Ak prebehne autentizácia úspešne, MULTOS odošle ID čipovej karty, ktorá sa autentizovala. Pre konkrétne ID sú potom v SQL databáze zvolené všetky jeho záznamy, vždy v spojení ID karty - jeden UHF tag. UHF tagy spojené s konkrétnym ID sú nasledovne usporiadané do poľa, ktoré je využité pri korektnom overovaní užívateľa.

`Python` program následne zapne UHF čítačku a dá jej príkaz na skenovanie ID UHF tagov. V stave skenovania je čítačka buď dovtedy, kým užívateľ nenaskenuje všetky potrebné UHF tagy, alebo kým nevyprší stanovený časový limit.

Tu sa program opäť dostáva do stavu čakania na priloženie JavaCard.

Meraná bola doba autentizácie pomocou implementovaného autentizačného protokolu bez načítania UHF tagov, keďže tento čas závisí takisto od ľudského faktoru. Čas bol meraný 3000 krát, výsledok je priemer týchto meraní, ide o **1.5244 s**. Čas autentizácie značne predlžuje manuálne volanie `garbage collectoru` na JavaCard. Volanie prebieha každú pätnástu autentizáciu a túto predlžuje približne o jednu sekundu. Toto zdržanie je započítané v priemernom čase autentizácie uvedenom vyššie.

Priemerný čas autentizácie bez použitia `garbage collectoru` je **1.3237 s**, spriemerovaný z 10 priebehov programu. Priemerné trvanie hlavných častí protokolu bez volania `garbage collectoru` je zobrazené v tab. 2.5. Zvyšok času je spotrebovaný zvyšnými operáciami prebiehajúcimi na RaspberryPi. Keďže neboli merané priamo časy procesoru, môžu sa tieto mierne líšiť v závislosti od momentálneho zaťaženia hardvéru.

Tab. 2.5:

proces	**čipová karta**	**čas**
zaslanie pola bajtov	JavaCard	77 ms
násobenie bodu na EC 256b generovanie náhodného 256b čísla 3DES šifrovanie	MULTOS	337 ms
3DES dešifrovanie hešovanie SHA1 3DES šifrovanie	JavaCard	613 ms
3DES dešifrovanie 2x hešovanie SHA1	MULTOS	212 ms

Záver

Úlohou bakalárskej práce bolo porovnať RFID technológie a na základe prevedeného prieskumu vybrať vhodnú technológiu na implementáciu viacatribútového autentizačného systému založeného na RFID technológií a čipových kartách. Práca sa skladá z teoretického úvodu a praktickej časti, rozdelenej na meranie a porovnanie hardvéru a na implentáciu autentizačného systému.

Teoretická časť oboznamuje čitateľa s problematikou RFID. Delí RFID systémy do podskupín, čím efektívne sprehľadňuje ich funkciu a využitie. Popísaná je štandardizácia na základe ISO a EPCglobal. Časť je venovaná i konkrétnym čipovým kartám MIFARE, ktoré boli využité na meranie v druhej, praktickej časti. Teoretická časť takisto popisuje možné využitie kryptografických primitív na vybraných čipových kartách s procesorom.

Praktická časť pracuje s prideleným hardvérom, pre najlepšie určenie vhodnej technológie pri implementácii autentizačného protokolu. Skúma LF, HF i UHF systém, zmienené sú i systémy, ktoré pre svoju funkčnosť vyžadujú kontakt s napájaním. Hlavnými bodmi praktického merania systémov boli dosah čítačiek, rýchlosť identifikácie a jej chybovosť.

Meraním bolo zistené, že LF technológia nie je k ďalšiemu využitiu vrámci bakalárskej práce vhodná, pretože LF čítačky majú štandardne veľmi nízky dosah a LF tagy majú minimálnu funkcionalitu, teda vysielať svoje ID.

Pri technológii HF je opäť dosah čítačky veľmi nízky. HF tagy však môžu obsahovať procesor, či kryptografický koprocesor a tak ponúkajú možnosti implementácie kryptografických primitív aj komplexnejších protokolov.

V HF časti je demonštratívne porovnaný čas identifikácie a autentizácie na čipových kartách MIFARE. Identifikácia bola meraná na karte MIFARE Ultralight, pričom spiemerovaný čas bol 25.909 ms. Autentizácia bola zrekoštruovaná a prevedená na doposiaľ neprelomenej čipovej karte MIFARE DESFire EV1, kedy výsledný čas bol 144,52 ms, čo robilo autentizáciu približne 6 krát pomalšou ako identifikáciu.

UHF technológia dosiahla pozitívne výsledky v oblasti merania dosahu čítačky, kedy dosah niektorých tagov pri maximálnej sile čítačky dosahoval desiatky metrov. Z toho dôvodu bola UHF technológia určená na použitie v autentizačnom systéme na pozíciu atribútov, ktorých úlohou bude vysielať svoje ID.

Hlavnou časťou bakalárskej práce je implementácia autentizačného protokolu. V protokole boli využité JavaCard, ako bezkontaktná užívateľská čipová karta, MULTOS, ako bezpečné úložisko tajných kľúčov a druhá strana autentizačného protokolu a nakoniec servery menežujúce celý protokol, a to autentizačný server alebo i terminál, ktorý sprostriedkováva autentizáciu a registračný server, ktorý generuje kľúče a identifikátory. Pre tieto bolo vybraté nenáročné zariadenie RaspberryPi, pričom ser-

very sú jednoducho prenesiteľné na akékoľvek iné zariadenie s operačným systémom založeným na Debian Linux. Podľa požiadavkov zadania bol navrhnutý viacatribútový autentizačný systém, ktorý berie na vedomie užívateľovo súkromie randomizovaním šifrovania jeho ID a využíva ako aj UHF tagy, tak aj čipové karty a nevýkonný počítač RaspberryPi2. Pre užívateľa je jednoducho použiteľný, po priložení JavaCard na čitačku a naskenovaní všetkých potrebných UHF tagov je užívateľ autentizovaný. Pre zaistenie bezpečnosti protokolu bola využitá kryptografia založená na eliptickej krivke NIST P256, čo znamená rýchlejšie výpočty a menšie využitie pamäte kariet.

Literatúra

[1] SMILEY, S.: *Active RFID vs. Passive RFID: What's the Difference?* [online]. 2016, posledná aktualizácia 4. 3. 2016 [cit. 2018-10-3]. Dostupné z URL: <https://blog.atlasrfidstore.com/active-rfid-vs-passive-rfid>

[2] *International Organization for Standardization* [online]. Dostupné z URL: <https://www.iso.org/home.html>

[3] *EPCglobal* [online]. Dostupné z URL: <https://www.gs1.org/epcglobal>

[4] CURTY, J., DECLERCQ, M., DEHOLLAIN, C., JOEHL, N.: *Design and Optimization of Passive UHF RFID Systems* 2007, 148 s. ISBN 0-387-35274-0.

[5] SMILEY, S.: *UHF RFID Frequency Regulations* [online]. 2014, posledná aktualizácia 2014-9-19 [cit. 2018-10-3]. Dostupné z URL: <https://blog.atlasrfidstore.com/uhf-rfid-frequency-regulations>

[6] FINKENZELLER, K.: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication *RFID Handbook, Third Edition*, 2012, 462 s. ISBN 978-0-470-69506-7.

[7] ARMSTRONG, S.: *Which RFID Frequency is Right for Your Application?* [online]. 2012, posledná aktualizácia 2012-10-29 [cit. 2018-12-8]. Dostupné z URL: <https://blog.atlasrfidstore.com/which-rfid-frequency-is-right-for-your-application>

[8] *AN11340, MIFARE Ultralight and MIFARE Ultralight EV1 Features and Hints*, 2018, 25 s.

[9] *ISO/IEC 7816-4, Identification cards* International Organization for Standardization, Second Edition, 2005-01-15, 83 s.

[10] *MF1S50YYX_V1, MIFARE Classic EV1 1K - Mainstream contactless smart card IC for fast and easy solution development*, NXP Semiconductors, revízia 3.2 z 2018-5-23, 33 s.

[11] *MF3ICDx21_41_81 MIFARE DESFire EV1 contavtless multi-application IC*, NXP Semiconductors, revízia 3.2 z 2015-12-9, 18 s.

[12] MINGYU, F., JINAHUA, W., GUANQWE I, W.: *A design of hardware cryptographic co-processor* [online]. Pridané do IEEE Xplore: 23. 9. 2003,[cit. 2018-10-26] Dostupné z URL: <https://ieeexplore.ieee.org/document/1232427>.

[13] *Types of smartcards* [online].[cit. 2018-11-11] Dostupné z URL: <http://www.smartcardbasics.com/smart-card-types.html>.

[14] LEHPAMER, H. *RFID Design Principles*, 2008, 293 s. ISBN 1-59693-194-9.

[15] ROBERTI, M.: *What Is a Semi-passive RFID Tag?* www.rfidjournal.com [online]. 01.10.2011 [cit. 2018-11-29]. Dostupné z URL: <https://www.rfidjournal.com/blogs/experts/entry?8117>

[16] BURDA, K. *Aplikovaná Kryptografie*, 2013, 255 s. ISBN 978-80-214-4612-0.

[17] LÓRENCZ, R.: *Bezpečnost* [online], 2011, [cit. 2018-11-30]. Dostupné z URL: <https://edux.fit.cvut.cz/oppa/BI-BEZ/prednasky/bez7.pdf>

[18] KALLA, T.: *Bezpečnost MIFARE karet*, bakalárska práca, Brno: Masarykova Univerzita, Fakulta Informatiky, 2012.

[19] FERGUSON, N., SCHNEIER, B.: *Practical Cryptography*, 2003, 410 s. ISBN 0-471-22357-3.

[20] POSPÍŠIL, R.: *Kryptografické algoritmy a faktorizace velkých čísel*, Bankovní institut vysoká škola Praha, Katedra matematiky, statistiky a informačních technologií, 2013. Vedoucí bakalářské práce Ing. Vladimír Beneš.

[21] GEYER, L.: *Eliptické křivky v kryptografii.* Brno: Vysoké učení technické v Brně, Fakulta eletrotechniky a komunikačních technlogií, 2010 34 s. Vedoucí bakalářské práce Ing. Petra Lambertová.

[22] HAJNÝ, J.: *Identifikace a autentizace v informačních systémech*, bakalárska práca, Praha: Bankovní institut vysoká škola Praha, Katedra informačních technologií a elektronického obchodování, 2019.

[23] Strnadel, P.: *Moderní kryptografie a systémy řízení přístupu*, Brno: Vysoké učení technické v Brně, [cit. 2018-11-29], 30 s.

[24] JUELS, A.: *RFID Security and Privacy: A Research Survey*, Invited Paper, IEEE, 0733-8716, 2006, 14 s.

[25] ALMEIDA, M.: *Hacking Mifare Classic Card* [online], 2014,[cit. 2018-12-1]. Dostupné z URL: <https://www.blackhat.com/docs/sp-14/materials/arsenal/sp-14-Almeida-Hacking-MIFARE-Classic-Cards-Slides.pdf>

[26] ROUSSEAU, L.: *pcsc-tools* [online], 15.4.2018, [cit. 18.11.2018]. Dostupné z URL: <http://ludovic.rousseau.free.fr/softwares/pcsc-tools/>

[27] *Package javax.smartcardio*, ORACLE [online], 2005, [cit. 1.12.2018]. Dostupné z URL: <https://docs.oracle.com/javase/7/docs/jre/api/security/smartcardio/spec/javax/smartcardio/package-summary.html>

[28] *UHF RFID Integrated Reader*, datasheet, 2 s.

[29] *uhfreader18* [online], 16.10.2018, [18.11.2018]. Dostupné z URL: <https://github.com/loblik/uhfreader18>

[30] *Cryptographic Key Length Recommendation* [online], [cit. 12.12.2018]. Dostupné z URL: <https://www.keylength.com/en/4/>

[31] *UBM21-Z Family* [online], [cit. 30.03.2019]. Dostupné z URL: <https://www.multos.com/products/approved_platforms/MIR/ubivelox/umb21>

[32] *JAVA CARD DEVELOPMENT KIT* [online], [cit. 30.03.2019]. Dostupné z URL: <https://www.oracle.com/technetwork/java/embedded/javacard/downloads/javacard-sdk-2043229.html>

[33] *JCAlgTest* [online], [cit. 30.03.2019]. Dostupné z URL: <https://github.com/crocs-muni/JCAlgTest>

[34] *Tools and SDK* [online], [cit. 31.03.2019]. Dostupné z URL: <https://www.multos.com/developer_centre/tools_and_sdk>

[35] *pyscard smartcard library for python* [online], [cit. 31.03.2019]. Dostupné z URL: <https://github.com/LudovicRousseau/pyscard>

[36] *An Introduction to Java Card Technology - Part 1* [online], [cit. 31.03.2019]. Dostupné z URL: <https://www.oracle.com/technetwork/java/javacard/javacard1-139251.html>

[37] *Writing a Java Card Applet* [online], [cit. 02.04.2019]. Dostupné z URL: <https://www.oracle.com/technetwork/java/javacard/intro-139322.html>

[38] HAJNÝ, J., DZURENDA, P., MALINA, L.: *Research Article Multidevice Authentication with Strong Privacy Protection* [online], Brno University of Technology, Czech Republic, 2018 [cit. 2018-12-02]. Dostupné z URL: <https://doi.org/10.1155/2018/3295148>

Zoznam skratiek

RFID	Radio Frequency Identification
UPC-A	Universal Product Code type A
AC	Alternating Current
API	Application Programming Interface
LF	Low Frequency
HF	High Frequency
UHF	Ultra High Frequency
ISO	International Organization of Standardization
IEC	International Electrotechnical Commission
EPCglobal	Electronic Product Code Global Incorporated
EPC	Electronic Product Code
APDU	Application Protocol Data Unit
CRC	Cyclic-Redundancy check
UID	Unique IDentity
ID	IDentifier
TID	Tag IDentifier
EAS	Electronic Article Surveillance
EEPROM	Electrically Erasable Programmable Read-Only Memory
ROM	Read-Only Memory
RAM	Random Access Memory
ECB	Electronic Code Book
CBC	Cypher Block Chaining
NFC	Near Field Communication
DES	Data Encryption Standard
3DES	Triple Data Encryption Standard
AES	Advanced Encryption Standard
RSA	Rivest–Shamir–Adleman cryptosystem
NIST	National Institute of Standards and Technology
ECDH	Elliptic Curve Diffie-Helman
ECDSA	Elliptic Curve Digital Signature Algorithm
GUI	Graphical User Interface
SAM	Secure Access Module
JCRE	JavaCard Runtime Environment
JRE	Java Runtime Environment
JCVM	JavaCard Virtual Machine
SDK	Software Development Kit
MULTOS	Multiple Operating System

ALU	Application Load Unit
ASN.1	Abstract Syntax Notation One

Zoznam príloh

A Obsah priloženého CD **65**

A Obsah priloženého CD

Priložené CD obsahuje všetky potrebné súbory pre prevázkovanie navrhnutého autentizačného systému. Súbor `registration.py` je hlavný skript pre registračný server, súbor `authentication.py` obsahuje hlavný program pre autentizáciu.

`Python` moduly `SQLModul.py`, `SmartCardModul.py` sú pomocné časti autentizácie a zároveň aj registrácie. Modul `UHFReader18.py` je upravený ovládač na UHF čítačku, ktorého originálna verzia je dostupná na githube[29].

V zložkách `javacard` a `multos` sú zdrojové kódy bežiace na čipových kartách a skompilované súbory nahrateľné na danú kartu.

Súbor `authentication.service` obsahuje configuráciu systemd servisy obsluhujúcej autentizačný proces.

Inštalácia a sprevádzkovanie autentizačného systému je popísané v `README.md`

Mimo autentizačného systému CD obsahuje súbor `DESFireAuth.java`, pre autentizáciu čipovej karty MIFARE DESFire.